REPORT DOCUMENTATION PAGE

Form Approved OMB NO. 0704-0188

1. REPORT DATE (DD-MM-YYYY)	2. REPORT TYPE	3. DATES COVERED (From - To)
21-03-2015	Ph.D. Dissertation	-

4. TITLE AND SUBTITLE
Adaptive modeling of details for physically-based sound synthesis and propagation

5a. CONTRACT NUMBER

5b. GRANT NUMBER
W911NF-13-C-0037

5c. PROGRAM ELEMENT NUMBER
665502

6. AUTHORS
Hengchin Yeh

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAMES AND ADDRESSES
Impulsonic, Inc.
305 Brookside Drive
Chapel Hill, NC 27516 -2905

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS (ES)
U.S. Army Research Office
P.O. Box 12211
Research Triangle Park, NC 27709-2211

10. SPONSOR/MONITOR'S ACRONYM(S)
ARO

11. SPONSOR/MONITOR'S REPORT NUMBER(S)
62557-CS-ST2.8

12. DISTRIBUTION AVAILIBILITY STATEMENT
Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES
The views, opinions and/or findings contained in this report are those of the author(s) and should not contrued as an official Department of the Army position, policy or decision, unless so designated by other documentation.

14. ABSTRACT
In order to create an immersive virtual world, it is crucial to incorporate a realistic aural experience that complements the visual sense. Physically-based sound simulation is a method to achieve this goal and automatically provides audio-visual correspondence. It simulates the physical process of sound: the pressure variations of a medium originated from some vibrating surface (sound synthesis), propagating as waves in space and reaching human ears (sound propagation). The perceived realism of simulated sounds depends on the accuracy of the computation methods and the computational resource available, and oftentimes it is not feasible to use the

15. SUBJECT TERMS
Applied sciences, Adaptive modeling, Physcially-based, Sound synthesis, Propagation, Virtual world

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	15. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Dinesh Manocha
UU	UU	UU	UU		19b. TELEPHONE NUMBER 919-590-6049

Report Title

Adaptive modeling of details for physically-based sound synthesis and propagation

ABSTRACT

In order to create an immersive virtual world, it is crucial to incorporate a realistic aural experience that complements the visual sense. Physically-based sound simulation is a method to achieve this goal and automatically provides audio-visual correspondence. It simulates the physical process of sound: the pressure variations of a medium originated from some vibrating surface (sound synthesis), propagating as waves in space and reaching human ears (sound propagation). The perceived realism of simulated sounds depends on the accuracy of the computation methods and the computational resource available, and oftentimes it is not feasible to use the most accurate technique for all simulation targets. I propose techniques that model the general sense of sounds and their details separately and adaptively to balance the realism and computational costs of sound simulations.

For synthesizing liquid sounds, I present a novel approach that generate sounds due to the vibration of resonating bubbles. My approach uses three levels of bubble modeling to control the trade-offs between quality and efficiency: statistical generation from liquid surface configuration, explicitly tracking of spherical bubbles, and decomposition of non-spherical bubbles to spherical harmonics. For synthesizing rigid-body contact sounds, I propose to improve the realism in two levels using example recordings: first, material parameters that preserve the inherent quality of the recorded material are estimated; then extra details from the example recording that are not fully captured by the material parameters are computed and added. For simulating sound propagation in large, complex scenes, I present a novel hybrid approach that couples numerical and geometric acoustic techniques. By decomposing the spatial domain of a scene and applying the more accurate and expensive numerical acoustic techniques only in limited regions, a user is able to allocate computation resources on where it matters most.

ADAPTIVE MODELING OF DETAILS FOR PHYSICALLY-BASED SOUND SYNTHESIS AND PROPAGATION

Hengchin Yeh

A dissertation submitted to the faculty of the University of North Carolina at Chapel Hill in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the Department of Computer Science.

Chapel Hill
2014

Approved by:

Gary Bishop

Ming C. Lin

Dinesh Manocha

Marc Niethammer

Nikunj Raghuvanshi

ABSTRACT

HENGCHIN YEH: Adaptive Modeling of Details for Physically-based Sound Synthesis and Propagation
(Under the direction of Ming C. Lin)

In order to create an immersive virtual world, it is crucial to incorporate a realistic aural experience that complements the visual sense. Physically-based sound simulation is a method to achieve this goal and automatically provides audio-visual correspondence. It simulates the physical process of sound: the pressure variations of a medium originated from some vibrating surface (sound synthesis), propagating as waves in space and reaching human ears (sound propagation). The perceived realism of simulated sounds depends on the accuracy of the computation methods and the computational resource available, and oftentimes it is not feasible to use the most accurate technique for all simulation targets. I propose techniques that model the general sense of sounds and their details separately and adaptively to balance the realism and computational costs of sound simulations.

For synthesizing liquid sounds, I present a novel approach that generate sounds due to the vibration of resonating bubbles. My approach uses three levels of bubble modeling to control the trade-offs between quality and efficiency: statistical generation from liquid surface configuration, explicitly tracking of spherical bubbles, and decomposition of non-spherical bubbles to spherical harmonics. For synthesizing rigid-body contact sounds, I propose to improve the realism in two levels using example recordings: first, material parameters that preserve the inherent quality of the recorded material are estimated; then extra details from the example recording that are not fully captured by the material parameters are computed and added. For simulating sound propagation in large, complex scenes, I present a novel hybrid approach that couples numerical and geometric acoustic techniques. By decomposing the spatial domain of a scene and applying the more accurate and expensive numerical acoustic techniques only in limited regions, a user is able to allocate computation resources on where it matters most.

To my parents

ACKNOWLEDGEMENTS

The many years that I spent working in the Department of Computer Science at UNC Chapel Hill on the research projects that would finally lead to this dissertation have been a great journey. First and foremost, I would like to thank my advisor, Prof. Ming C. Lin, for her guidance and support, as well as the freedom that she granted me in exploring my research interest, and patience that she showed for every change of my research directions. I would also like to thank the members of my committee, Prof. Gary Bishop, Prof. Dinesh Manocha, Prof. Marc Niethammer, and Dr. Nikunj Raghuvanshi, for tgation.

I would also like to thank the collaborators I have had the honor of working with, including Zhimin Ren, Ravish Mehra, Lakulish Antani, Sean Curtis, Qi Mo, Will Moss, Jur van den Berg, and Sachin Patil. I would also like to thank Dr. Anish Chandak at Impulsonic Incorporation and Dr. Micah Taylor at Rose-Hulman Institute of Technology for their insight and help, and Dr. J. Rafael Tena at Disney Research for his guidance as my intern supervisor. I would also like to thank the many anonymous reviewers for helping me improve the quality of my work.

The most I owe is to my parents, Hsiu-Lin Tsao and Song-So Yeh, for your unconditional support and encouragement, as well as putting up with all the years that I could not go home and be with you. To my many friends at Chapel Hill, including Chih-Da Wu, Ching-Ching Lin, Jui-Hua Hsieh, Aleks Sedykh, Huai-Ping Lee, Chien-Yi Hou, and Ya-Chun Li: I would not have made this far without your company and support. Finally, I would like to thank Yu-Chi Chen, who was there through the toughest part of my PhD years till the end. Thank you.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1: INTRODUCTION

In our real-world experience, we are constantly submerged in a wide variety of sounds. The *aural* experience complements the visual sense. For example, when we see a wave crashing on a beach we expect to hear the splashing sound. When we walk toward talking people we expect to hear them more clearly, and the voice should become less distinctive when we walk around a corner. In a virtual environment, being able to incorporate sound effects that corresponds to visual events greatly enhances users' immersion. Sound effect production thus has a wide application in video games, computer animation, films, training systems, computer aided design, scientific visualization, and assistive technology for the visually impaired.

Traditional methods of incorporating sound effect is a laborious practice. talented Foley artists are normally employed to record a large number of sound samples in advance and manually edit and synchronize the recorded sounds to a visual scene. This approach generally achieves satisfactory results. However, it is labor-intensive and cannot be applied to all interactive applications. It is still challenging, if not infeasible, to produce sound effects that precisely capture complex interactions that cannot be predicted in advance.

Therefore *physically-based sound simulation* has been developed as a method to automatically integrate sounds into a virtual environment. It aims to simulate the physical process of sound, which is essentially the pressure variations of a medium originated from some vibration of surface, propagating in space and reaching human ears. Recent progress has been made on sound synthesis models that automatically produce sounds for various types of objects and phenomena. The practice directly provides audio-visual correspondence – it generates sounds that automatically synchronize with visual events and naturally capture the variation of object interactions (e.g. a ball bouncing or rolling, water in a brook running rapidly or calmly) or acoustic effects (e.g. the muffling of sound when the source is occluded from the listener).

Besides audio-visual correspondence, another factor is the quality of audio. In theory, if the perfect model of a physical phenomenon exists and infinite computing power is available, the resulting sound can be faithfully simulated from first principles. In practice, one model does not fit all. In some cases the existing model is not complete. For example, a universal damping model that can explain the vibration and sound-generating behavior of all materials is still an open research problem. In some cases the fine-scale dynamics is not resolved, especially when sound is to be generated from existing visual simulation. For example the fluid simulation in games usually provides only the surface information, and only in a coarse time resolution (30-60 fps). Even if an accurate model exists and all scales are resolved, the computational cost might be prohibitively high. On the other hand, simply omitting details and applying only coarse approximation often produces unsatisfactory results. Human ears are extremely sensitive to details: the 'crisp' noise of placing a coffee cup on a plate, the subtle variation of each rain drop, the acoustical quality of a concert hall – all contribute to perceived realism. A poorly simulated audio sounds 'fake' and affects the sense of immersion.

1.1 Adaptive Modeling of Details

In order to efficiently produce faithful aural experience for a complex sound source or environment, I propose techniques that model the general sense of sounds and their details separately. The principle is to first employs simplified, efficient methods to produce sounds that coarsely approximate the simulated sound sources (e.g. water motion, solid objects collision) or give a rough sense of the environment (e.g. a room or an open scene). Then rich and complex details are modeled separately and coupled into the system to improve realism of generated audios in an adaptive, user-controllable manner. The goal of my thesis is to develop simulation approaches that follow this general principle for many sound-related problems that are of interest to virtual environment applications.

For synthesizing liquid sounds, we adaptively model bubbles in different levels of details, because the dominant source of sound generated by liquid is the oscillation of bubbles within the fluid medium. Given just the geometry and velocity of a water surface, liquid sounds can be simulated in real time through statistical bubble generation and radius distribution models. If bubbles are explicitly modeled and tracked, more faithful liquid sounds can be generated. Even more sound details can be added by considering non-spherical bubbles, where the shape deviation from a perfect sphere is

decomposed into spherical harmonics, and the sound from each harmonic is summed. By choosing which bubbles are statistically generated, which bubbles are explicitly tracked, and which bubbles' shapes are decomposed to spherical harmonics (and to what order), a user can control the trade-offs between realism and computational cost.

For synthesizing rigid-body contact sounds, linear modal synthesis is a powerful tool to simulate rigid-body sound in a physically-based manner, but the synthesized sounds are not as rich and realistic as real-world recordings. Recorded sounds, on the other hand, include a lot of details that linear modal synthesis does not model, such as fine-scale inhomogeneity, nonlinear resonant modes, and transient noise of unknown nature, are are still widely used in movies, animations, and games. I propose to improve the realism of linear modal synthesis in two levels. First, using an example recording to estimate the material parameters allows modal-synthesized sounds to preserve the inherent quality of the recorded material. Secondly, the difference between the example recording and the modal-synthesized sound is computed, transfered to different geometries if necessary, and added back to the final synthesized sound.

For simulating sound propagation in a large scene, the adaptive modeling of details is achieved by combining two different acoustic techniques. Traditionally, numerical acoustic technique are used to accurately model wave phenomena such as diffraction, interference, and scattering, but these techniques are generally expensive. Performing an accurate wave simulation for the entire scene, however, is usually not necessary – sound wave traveling in empty space and reflecting from large objects can be more efficiently modeled as rays with geometric acoustic techniques. Only in the vicinity of objects smaller than the wavelength of the sound waves are the wave phenomena significant and numerical techniques required. I propose to decompose the spatial domain of a scene and apply the numerical acoustic techniques only in limited, smaller regions, allowing a user to allocate computation resources on where it matters the most.

1.2 Thesis Statement

Realistic sounds from complex physical systems such as liquids and rigid bodies, as well as propagation in a large scene, can be efficiently simulated on current hardware through physically-based sound synthesis and propagation techniques that model details separately and adaptively.

1.3 Challenges and Contributions

My contributions can be divided into three main areas, the simulation of liquid sounds, rigid-body contact sounds, and sound propagation. I will discuss the respective computational challenges as well as my contributions.

1.3.1 Sound Simulation from Fluid Simulation

I investigate new methods for sound synthesis in a liquid medium in the first part of my thesis. Our formulation is based on prior work in physics and engineering, which shows that sound is generated by the resonance of bubbles within the fluid (Rayleigh, 1917). We couple physics-based fluid simulation with the automatic generation of liquid sound based on Minneart's formula (Minnaert, 1933) for spherical bubbles and spherical harmonics (Leighton, 1994) for non-spherical bubbles. We also present a fast, general method for tracking the bubble formations and a simple technique to handle a large number of bubbles within a given time budget.

The proposed synthesis algorithm offers the following advantages:

- It renders both liquid sounds and visual animation simultaneously using the same fluid simulator.
- It introduces minimal computational overhead on top of the fluid simulator.
- For fluid simulators that generates bubbles, no additional physical quantities, such as force, velocity, or pressure are required – only the geometry of bubbles.
- For fluid simulators without bubble generation, a physically-inspired bubble generation scheme provides plausible audio.
- It can adapt and balance between computational cost and quality.

We also decouple sound rendering rates (44,000 Hz) from graphical updates (30-60 Hz) by distributing the bubble processing over multiple audio frames.

1.3.2 Example-Guided Rigid Body Sound Synthesis

In real-time applications, *modal synthesis* methods are often used for simulating sounds. This approach generally does not depend on any pre-recorded audio samples to produce sounds triggered by all types of interactions, so it does not require manually synchronizing the audio and visual events. The produced sounds are capable of reflecting the rich variations of interactions and also the geometry of the sounding objects. Although this approach is not as demanding during run time, setting up good initial parameters for the virtual sounding materials in *modal analysis* is a time-consuming and non-intuitive process. For a complicated scene consisting of many different sounding materials, the parameter selection procedure can quickly become prohibitively expensive and tedious.

Although tables of material parameters for stiffness and mass density are widely available, directly looking up these parameters in physics handbooks does not offer intuitive, direct control as using a recorded audio example. In fact, sound designers often record their own audio to obtain the desired sound effects. This chapter presents a new data-driven sound synthesis technique that preserves the realism and quality of audio recordings, while exploiting all the advantages of physically based modal synthesis. We introduce a computational framework that takes just one example audio recording and estimates the intrinsic *material parameters* (such as stiffness, damping coefficients, and mass density) that can be directly used in modal analysis.

As a result, for objects with different geometries and run-time interactions, different sets of modes are generated or excited differently, and different sounds are produced. However, if the material properties are the same, they should all sound like coming from the same material. For example, a plastic plate being hit, a plastic ball being dropped, and a plastic box sliding on the floor generate different sounds, but they all sound like 'plastic', as they have the same material properties. Therefore, if we can deduce the material properties from a recorded sound and *transfer* them to different objects with rich interactions, the *intrinsic quality* of the original sounding material is preserved. Our method can also compensate the differences between the example audio and the modal-synthesized sound. Both the material parameters and the residual compensation are capable of being transfered to virtual objects of varying sizes and shapes and capture all forms of interactions.

The key contributions of my approach are summarized below:

- A feature-guided parameter estimation framework to determine the optimal material parameters that can be used in existing modal sound synthesis applications.
- An effective residual compensation method that accounts for the difference between the real-world recording and the modal-synthesized sound.
- A general framework for synthesizing rigid-body sounds that closely resemble recorded example materials.
- Automatic transfer of material parameters and residual compensation to different geometries and runtime dynamics, producing realistic sounds that vary accordingly.

1.3.3 Wave-Ray Hybrid Sound Propagation

Sound propagation techniques are used to model how sound waves travel in the space and interact with various objects in the environment. Sound propagation algorithms are used in many interactive applications, such as computer games or virtual environments, and offline applications, such as noise prediction in urban scenes, architectural acoustics, virtual prototyping, etc.. Realistic sound propagation that can model different acoustic effects, including diffraction, interference, scattering, and late reverberation, can considerably improve a user's immersion in an interactive system and provides spatial localization (Blauert, 1983).

The acoustic effects can be accurately simulated by numerically solving the acoustic wave equation. Some of the well-known solvers are based on the boundary-element method, the finite-element method, the finite-difference time-domain method, etc. However, the time and space complexity of these solvers increases linearly with the volume of the acoustic space and is a cubic (or higher) function of the source frequency. As a result, these techniques are limited to interactive sound propagation at low frequencies (e.g. 1-2KHz) (Raghuvanshi et al., 2010; Mehra et al., 2013), and may not scale to large environments.

Many interactive applications use geometric sound propagation techniques, which assume that sound waves travels like rays. This is a valid assumption when the sound wave travels in free space or when the size of intersecting objects is much larger than the wavelength. As a result, these geometric techniques are unable to simulate many acoustic effects at low frequencies, including diffraction,

interference, and higher-order wave effects. Many hybrid combinations of numeric and geometric techniques have been proposed, but they are limited to small scenes or offline applications.

I have developed a novel hybrid approach that couples geometric and numerical acoustic techniques to perform interactive and accurate sound propagation in complex scenes. My approach uses a combination of spatial decomposition and frequency decomposition, along with a novel two-way wave-ray coupling algorithm. The entire simulation domain is decomposed into different regions, and the sound field is computed separately by geometric and numerical techniques for each region. In the vicinity of objects whose sizes are comparable to the simulated wavelength (near-object regions), we use numerical wave-based methods to simulate all wave effects. In regions away from objects (far-field regions), including the free space and regions containing objects that are much larger than the wavelength, we use a geometric ray-tracing algorithm to model sound propagation. We restrict the use of numeric propagation techniques to small regions of the environment and precompute the pressure field at low frequencies. The rest of the pressure field is precomputed using ray tracing.

At the interface between near-object and far-field regions, we need to couple the pressures computed by the two different (one numerical and one geometric) acoustic techniques. Rays entering a near-object region define the incident pressure field that serves as the input to the numerical acoustic solver. The numerical solver computes the outgoing scattered pressure field, which in turn has to be represented by rays exiting the near-object region. At the core of our hybrid method is a two-way coupling procedure that handles these cases. We present a scheme that represents two-way coupling using *transfer functions* and computes all orders of interaction.

The key results of my work include:

- *An efficient hybrid approach* that decomposes the scene into regions that are more suitable for either geometric or numerical acoustic techniques, exploiting the strengths of both.

- *Novel two-way coupling between wave-based and ray-based acoustic simulation* based on fundamental solutions at the interface that ensures the consistency and validity of the solution given by the two methods. Transfer functions are used to model two-way couplings to allow multiple orders of acoustic interactions.

- *Fast, memory-efficient interactive audio rendering* that only uses tens to hundreds of megabytes of memory.

We have also tested our technique on a variety of scenarios and integrated our system with the Valve's Source™game engine. Our technique is able to handle both large indoor and outdoor scenes (similar to geometric techniques) as well as generate realistic acoustic effects (similar to numeric wave solvers), including late reverberation, high-order reflections, reverberation coloration, sound focusing, and diffraction low-pass filtering around obstructions. Furthermore, our pressure evaluation takes orders of magnitude less memory compared to state-of-the-art wave equation solvers.

1.4 Thesis Organization

The following chapters are organized as follows. In the next chpater, I discuss related work in the areas of sound synthesis (for liquid sounds and rigid body sounds) and sound propagation. Then, three chapters are devoted to describe the three main key contributions of my thesis work: sound synthesis from fluid simulation, example-guided rigid body sound synthesis, and wave-ray hybrid sound propagation. I conclude my thesis with a summary of the main results, as well as a discussion of future work.

CHAPTER 2: PREVIOUS WORK

In this chapter I review related work in sound synthesis and sound propagation.

2.1 Sound Synthesis

In the last couple of decades, there has been strong interest in digital sound synthesis in both computer music and computer graphics communities due to the needs for auditory display in virtual environment applications. The traditional practice of Foley sounds is still widely adopted by sound designers for applications like video games and movies. Real sound effects are recorded and edited to match a visual display. More recently, *granular synthesis* became a popular technique to create sounds with computers or other digital synthesizers. Short grains of sounds are manipulated to form a sequence of audio signals that sound like a particular object or event. Roads (2004) gave an excellent review on the theories and implementation of generating sounds with this approach. Picard et al. (2009) proposed techniques to mix sound grains according to events in a physics engine.

Another approach for simulating sound sources is *physically based sound synthesis*. Sounds of interesting natural phenomena as well as object interactions are simulated from physical principles, and the synthesized sounds automatically synchronize with the visual rendering. My work on sound synthesis follows this approach. I review the related work of physically-based simulation of liquid and rigid-body sounds, as well as work on improving realism of synthesized sound by acquiring parameters from real audio recordings and incorporating residuals.

2.1.1 Liquid Sounds

Since the seminal works of Foster and Metaxas (1996), Stam (1999), and Foster and Fedkiw (2001), there has been tremendous interest and research on *visual simulation* of fluids in computer graphics. Generally speaking, current algorithms for visual simulation of fluids can be classified into three broad categories: grid-based methods, smoothed particle hydrodynamics (SPH), and

shallow-water approximations. We refer the reader to a recent survey (Bridson and Müller-Fischer, 2007) for more details.

For *audio simulation*, the physics literature presents extensive research on the acoustics of bubbles, dating back to the work of Lord Rayleigh (1917). There have been many subsequent efforts, including works on bubble formation due to drop impact (Pumphrey and Elmore, 1990; Prosperetti and Oguz, 1993) and cavitation (Plesset and Prosperetti, 1977), the acoustics of a bubble popping (Ding et al., 2007), as well as multiple works by Longuet-Higgins presenting mathematical formulations for monopole bubble oscillations (1989b; 1989a) and non-linear oscillations (1991). T. G. Leighton's (1994) excellent text covers the broad field of bubble acoustics and provides many of the foundational theories for my work.

Van den Doel (2005) introduced the first method in computer graphics for generating liquid sounds. Using Minneart's formula, which defines the resonant frequency of a spherical bubble in an infinite volume of water in terms of the bubble's radius, van den Doel provides a simple technique for generating fluid sounds through the adustment of various parameters. Other previous liquid sound synthesis methods provide limited physical basis for the generated sounds (Imura et al., 2007). Zheng and James integrated fluid simulation with bubble-based sound synthesis to automatically generate liquid sounds (2009). They consider spherical bubbles as in (van den Doel, 2005), and focus on the propagation of sound – both from the bubble to the water surface and the water surface to the listener. Their numerical sound propagation is compute-intensive and requires tens of hours of compute time on a cluster.

A related topic is simulating sound generated by air movement, which is also governed by fluid dynamics. Previous works include sound resulting from objects moving rapidly through air (2003) and the sound of woodwinds and other instruments (Florens and Cadoz, 1991; Scavone and Cook, 1998). Sound generated by the turbulent field due to fire has also been simulated (Dobashi et al., 2004; Chadwick and James, 2011).

2.1.2 Rigid Body Sounds

Rigid-body sounds play a vital role in all types of virtual environments. O'Brien et al. (2001) proposed simulating rigid bodies with deformable body models that approximates solid objects' small-scale vibration leading to variation in air pressure, which propagates sounds to human ears.

Their approach accurately captures surface vibration and wave propagation once sounds are emitted from objects. However, it is far from being efficient enough to handle interactive applications. Adrien (1991) introduced *modal synthesis* to digital sound generation. For real-time applications, *linear modal sound synthesis* has been widely adopted to synthesize rigid-body sounds (van den Doel and Pai, 1998; O'Brien et al., 2002; Raghuvanshi and Lin, 2006; James et al., 2006a; Zheng and James, 2010). This method acquires a modal model (i.e. a bank of damped sinusoidal waves) using *modal analysis* and generates sounds at runtime based on excitation to this modal model. Moreover, sounds of complex interaction can be achieved with modal synthesis. Van den Doel et al. (2001) presented parametric models to approximate contact forces as excitation to modal models to generate impact, sliding, and rolling sounds. Ren et al. (2010) proposed including normal map information to simulate sliding sounds that reflect contact surface details.

More recently, Zheng and James (2011) created highly realistic contact sounds with linear modal synthesis by enabling non-rigid sound phenomena and modeling vibrational contact damping. The use of linear modal synthesis is not limited to creating simple rigid-body sounds. Chadwick et al. (2009) used modal analysis to compute linear mode basis, and added nonlinear coupling of those modes to efficiently approximate the rich thin-shell sounds. Zheng and James (2010) extended linear modal synthesis to handle complex fracture phenomena by precomputing modal models for ellipsoidal sound proxies. Moreover, the standard modal synthesis can be accelerated with techniques proposed by (Raghuvanshi and Lin, 2006; Bonneel et al., 2008), which make synthesizing a large number of sounding objects feasible at interactive rates.

However, few previous sound synthesis work addressed the issue of how to determine material parameters used in modal analysis to more easily recreate realistic sounds.

2.1.2.1 Parameter Acquisition

Spring-mass (Raghuvanshi and Lin, 2006) and finite element (O'Brien et al., 2002) representations have been used to calculate the modal model of arbitrary shapes. Challenges lie in how to choose the material parameters used in these representations. Pai et al. (2001) and Corbett et al. (2007) directly acquires a modal model by estimating modal parameters (i.e. amplitudes, frequencies, and dampings) from measured impact sound data. A robotic device is used to apply impulses on a real object at a large number of sample points, and the resulting impact sounds are analyzed for modal

parameter estimation. This method is capable of constructing a virtual sounding object that faithfully recreates the audible resonance of its measured real-world counterpart. However, each new virtual geometry would require a new measuring process performed on a real object that has exactly the same shape, and it can become prohibitively expensive with an increasing number of objects in a scene. This approach generally extracts hundreds of location-dependent parameters for one object from many audio clips, while the goal of our technique instead is to estimate only a few parameters that best represent one *material* of a sounding object from only *one* audio clip.

To the best of my knowledge, the only other research work that attempts to estimate sound parameters from one recorded clip is by Lloyd et al. (2011). Pre-recorded real-world impact sounds are utilized to find peak and long-standing resonance frequencies, and the amplitude envelopes are then tracked for those frequencies. They proposed using the tracked time-varying envelope as the amplitude for the modal model, instead of the standard damped sinusoidal waves in conventional modal synthesis. Richer and more realistic audio is produced this way. Their data-driven approach estimates the modal parameters instead of material parameters. Similar to the method proposed by Pai et al. (2001), these are per-mode parameters and not transferable to another object with corresponding variation. At runtime, they randomize the gains of all tracked modes to generate an illusion of variation when hitting different locations on the object. Therefore, the produced sounds do not necessarily vary correctly or consistently with hit points. Their adopted resonance modes plus residual resynthesis model is very similar to that of SoundSeed Impact (Audiokinetic, 2011), which is a sound synthesis tool widely used in the game industry. Both of these works extract and track resonance modes and modify them with signal processing techniques during synthesis. None of them attempts to fit the extracted data (which are pre-object based) to estimate a higher-level *per-material based* model.

In computer music and acoustic communities, researchers proposed methods to calibrate physically based virtual musical instruments. For example, Välimäki et al. (1996; 1997) proposed a physical model for simulating plucked string instruments. They presented a parameter calibration framework that detects pitches and damping rates from recorded instrument sounds with signal processing techniques. However, their framework only fits parameters for strings and resonance bodies in guitars, and it cannot be easily extended to extract parameters of a general rigid-body sound synthesis model. Trebian and Oliveira (2009) presented a sound synthesis method with linear

digital filters. They estimated the parameters for recursive filters based on pre-recorded audio and re-synthesized sounds in real time with digital audio processing techniques. This approach is not designed to capture rich physical phenomena that are automatically coupled with varying object interactions. The relationship between the perception of sounding objects and their sizes, shapes, and material properties have been investigated with experiments, among which Lakatos et al. (1997) and Fontana (2003) presented results and studied human's capability to tell materials, sizes, and shapes of objects based on their sounds.

2.1.2.2 Modal Plus Residual Models

The sound synthesis model with a deterministic signal plus a stochastic residual was introduced to spectral synthesis by Serra and Smith (1990). This approach analyzes an input audio and divides it into a deterministic part, which are time-variant sinusoids, and a stochastic part, which is obtained by spectral subtraction of the deterministic sinusoids from the original audio. In the resynthesis process, both parts can be modified to create various sound effects as suggested by Cook (1996; 1997; 2002) and Lloyd et al. (2011). Methods for tracking the amplitudes of the sinusoids in audio dates back to Quateri and McAulay (1985), while more recent work (Serra and Smith III, 1990; Serra, 1997; Lloyd et al., 2011) also proposes effective methods for this purpose. All of these works directly construct the modal sounds with the extracted features. In contrast, our modal component is synthesized with the estimated material parameters. Therefore, although I adopt the same concept of modal plus residual synthesis for our framework, I face very different constraints due to the new objective in material parameter estimation, and render these existing works not applicable to the problem addressed in my thesis.

2.2 Sound Propagation

Computational acoustics studies the propagation of sound through a medium and may be roughly classified into Geometric Acoustics and Numerical Acoustics depending on how wave propagation is modeled. There has also been effort to combine the two techniques.

2.2.1 Numerical Acoustic Techniques

Accurate, numerical acoustic simulations typically solve the acoustic wave equation using numerical methods. The Finite Difference Time Domain (FDTD) method was originally proposed to model electromagnetic waves (Yee, 1966; Taflove and Hagness, 2005). It discretizes space as a uniform grid and solves for the field values at each cell for discrete time steps. It has been an adopted to room acoustics problems (Botteldooren, 1994, 1995) and has recently been applied to medium sized 3D scenes (Sakamoto et al., 2002, 2004, 2006). The Finite Element Method (FEM) (Zienkiewicz et al., 2006; Thompson, 2006) and the Boundary Element Method (BEM) (Cheng and Cheng, 2005; Gumerov and Duraiswami, 2009) discretize the scene's volume and surface into elements respectively. They are usually employed to solve the steady-state frequency domain response, with FEM applied mainly to interior and BEM to exterior acoustic problems (Kleiner et al., 1993). Digital Waveguide Mesh approaches (Van Duyne and Smith, 1993) roots in musical synthesis and use discrete waveguide elements to propagate acoustic waves along a single dimension (Savioja, 1999; Karjalainen and Erkut, 2004; Murphy et al., 2007). Recently Raghuvanshi et al. proposed a method based on adaptive rectangular decomposition (2009a). It achieves high accuracy with a coarse spatial discretization.

These techniques, however, require the volume or boundary of the scene to be discretized at least twice the Nyquist frequency, and their time and space complexity increases as a third or fourth power of frequencies. Hence, these techniques often require many hours of simulation time and gigabytes of storage to model low frequencies in large scenes with static sources, and they scale as the third or fourth power of frequency. Despite recent advances, they remain impractical for many real-time applications.

Equivalent source method, also called the Method of Fundamental solutions (Ochmann, 1995, 1999), expresses the solution fields of the wave equation in terms of a linear combination of points sources of various order (monopoles, dipoles, etc). The main idea behind this technique is to choose the positions and amplitudes of these elementary sources such that the boundary condition is satisfied. Thus, the resulting solution satisfies the wave equation. Recently, Mehra et al. (2013) proposed a novel sound propagation technique for large outdoor scenes based on equivalent sources. James et al. (2006b) solved a related *sound radiation* problem, using equivalent sources to represent the radiation field generated by a vibrating object.

2.2.2 Geometric Acoustic Techniques

Most acoustics simulation software and commercial systems are based on geometric techniques (Funkhouser et al., 1998; Vorlander, 1989) that assume sound travels along linear rays (Funkhouser et al., 2004). These methods are often based on stochastic ray tracing (Vorlander, 1989) or image sources (Borish, 1984). They frequently take advantage of recent advances in CPU- and/or GPU-based ray tracing techniques (Taylor et al., 2009, 2012) or frustum tracing (Chandak et al., 2008; Lauterbach et al., 2007) to efficiently approximate sound propagation in complex, dynamic scenes. The simplified assumption of rays limits these methods to accurately capture specular and diffuse reflections only at high frequencies. Diffraction is typically modeled by identifying individual diffracting edges (Svensson et al., 1999; Tsingos et al., 2001). These ray-based techniques can interactively model early reflections and first order edge-diffraction (Taylor et al., 2012); however, they cannot interactively model the reverberation of the impulse response explicitly, since that would require high-order reflections and wave effects such as scattering, interference, and diffraction. Hence, many commercial systems approximate reverberation using the parameters of simple statistical models (Eyring, 1930).

While ray-tracing has been successfully used in many interactive acoustics systems (Lentz et al., 2007), the number of rays traced has to be limited for scenes with moving listeners in order to maintain real-time performance. As the worst-case complexity of image source methods scales exponentially with the number of polygons in the scene, some interactive systems often group the polygons to simplify the scene representation (Alarcao et al., 2010; Joslin and Magnenat-Thalmann, 2003).

2.2.3 Hybrid Techniques

Several methods for combining geometric and numerical acoustic techniques have been proposed. One line of work is based on *frequency decomposition*: dividing the frequencies to be modeled into low and high frequencies. Low frequencies are modeled by numerical acoustic techniques, and high frequencies are treated by geometric methods, including the finite difference time domain method (FDTD) (Southern et al., 2011; Lokki et al., 2011), the digital waveguide mesh method (DWM) (Murphy et al., 2008), and the finite element method (FEM) (Granier et al., 1996; Aretz,

2012). However, these methods use numerical methods at lower frequencies over the entire domain. As a result, they are limited to offline applications and may not scale to very large scenes.

Another method of hybridization is based on *spatial decomposition*. The entire simulation domain is decomposed to different regions: near-object regions are handled by numerical acoustic techniques to simulate wave effects, while far-field regions are handled by geometric acoustic techniques. Hampel et al. (2008) combine the boundary element method (BEM) and geometric acoustics using a spatial decomposition. Their method provides a one-way coupling from BEM to ray tracing, converting pressures in the near-object region (computed by BEM) to rays that enter the far-field region containing the listener. In electromagnetic wave propagation, Wang et al. (2000) propose a hybrid technique combining ray tracing and FDTD. Their technique is also based on a one-way coupling, where rays are traced in the far-field region and collected at the boundaries of the near-object regions. The pressures are then evaluated and serve as the boundary condition for the FDTD method. These one-way coupling methods do not allow rays to enter and exit the near-object regions of an object, and therefore acoustic effects of that object will not be propagated to the far-field regions. Barbone et al. (1998) propose a two-way coupling that combines the acoustic field generated using ray-tracing and FEM. Jean et al. (2008) present a hybrid BEM/beam tracing approach to compute the radiation of tyre noise. However, these methods do not describe how multiple entrance of rays into near-object regions of different objects is handled, which is crucial when simulating interaction between multiple objects.

2.2.4 Acoustic Kernel-Based Interactive Techniques

There has been work in enabling interactive auralization for acoustic simulations through precomputation. At a high level, these techniques tend to precompute an acoustic kernel, which is used at runtime for interactive propagation in static environments. Raghuvanshi et al.(2010) precompute acoustic responses on a sampled spatial grid using a numerical solver. They then encode perceptually salient information to perform interactive sound rendering. Mehra et al. (2013) proposed an interactive sound propagation technique for large outdoor scenes based on equivalent sources. Other techniques use geometric methods to precompute high-order reflections or reverberation (Tsingos, 2009; Antani et al., 2012) and compactly store the results for interactive sound propagation at runtime. Our method

can be integrated into any of these systems as an acoustic kernel that can efficiently capture wave effects in a large scene.

CHAPTER 3: SOUND SYNTHESIS FROM FLUID SIMULATION

In this chapter, I discuss my work on performing sound synthesis from fluid simulation. The rest of this chapter is organized as follows – in the next section I describe the physical principles of liquid sound. After that, I describe how liquid sound can be simulated by integrating various kinds of fluid simulators. Following this, I discuss the implementation details and the results obtained with my approach. Finally I conclude with a summary of my contributions and a discussion of limitations of my approach and possible directions of future work.

3.1 Liquid Sound Principles

Sound is produced by surface vibrations of an object under force(s). These vibrations travel through the surrounding medium to the human ear and the changes in pressure are perceived as sound. In the case of fluids, sound is primarily generated by bubble formation and resonance, creating pressure waves that travel though both the liquid and air media to the ear. Although an impact between a solid and a liquid will generate some sound directly, the amplitude is far lower than the sound generated from the created bubbles. We refer the reader to Leighton's (1994) excellent text on bubble acoustics for more detail, and present an overview of the key concepts below.

3.1.1 Spherical Bubbles

Minneart's formula, which derives the resonant frequency of a perfectly spherical bubble in an infinite volume of water from the radius, provides a physical basis for generating sound in liquids. Since external sound sources rarely exist in fluids and the interactions between resonating bubbles create a minimal effect while greatly increasing the computational cost, we assume that a bubble is given an initial excitation and subsequently oscillates, but is not continuously forced. The sound generated by the bubble will, therefore, be dominated by the resonant frequency, as other frequencies will be of lower magnitude and will rapidly die out after the bubble is created. Therefore, a resonating bubble acts like a simple harmonic oscillator, making the resonant frequency dependent on the

stiffness of the restoring force and the effective mass of the gas trapped within the bubble. The stiffness of the restoring force is the result of the pressure within the bubble and the effective mass is dependent on the volume of the bubble and the density of the medium. If we approximate the bubble as a sphere with radius, r_0, then for cases where $r_0 > 1\mu m$, the force depends predominantly on the ambient pressure of the surrounding water, p_0, and the resonant frequency is given by Minneart's formula,

$$f_0 = \frac{1}{2\pi}\sqrt{\frac{3\gamma p_0}{\rho r_0^2}}, \tag{3.1}$$

where γ is the specific heat of the gas ($\approx$ 1.4 for air), p_0 is the gas pressure inside the bubble at equilibrium (i.e. when balanced with the pressure of the surrounding water) and ρ the density of the surrounding fluid. For air bubbles in water, Equation 3.1 reduces to a simple form: $f_0 r_0 \approx 3m/s$. The human audible range is 20 Hz to 20 kHz, so we will restrict our model to the corresponding bubbles of radii, 0.15 mm to 15 cm.

An oscillating bubble, just like a simple harmonic oscillator, is subject to viscous, radiative, and thermal damping. Viscous damping rapidly goes to zero for bubbles of radius greater than 0.1 mm, so we will only consider thermal and radiative damping. We refer the reader to Section 3.4 of (Leighton, 1994) for a full derivation, and simply present the peritinant equations here. Thermal damping is the result of energy lost due to conduction between the bubble and the surrounding liquid, whereas radiative damping results from energy radiated away in the form of acoustic waves. These two can be approximated as,

$$\delta_{th} = \sqrt{\frac{9(\gamma-1)^2}{4G_{\text{th}}} f_0} \qquad \delta_{rad} = \sqrt{\frac{3\gamma p_0}{\rho c^2}}, \tag{3.2}$$

where c is the speed of sound and G_{th} is a dimensionless constant associated with thermal damping. The total damping is simply the sum, $\delta_{tot} = \delta_{th} + \delta_{rad}$.

Modeling the bubble as a damped harmonic oscillator, oscillating at Minneart's frequency, the impulse response is given by

$$p(t) = A_0 sin(2\pi f(t)t)e^{-\beta_0 t}, \tag{3.3}$$

where A_0 is determined by the initial excitation of the bubble and $\beta_0 = \pi f_0 \delta_{tot}$ is the rate of decay due to the damping term δ_{tot} given above. For single-mode bubbles in low concentration, We replace f_0 in

the standard harmonic oscillator equation with $f(t)$, where $f(t) = f_0(1 + \xi\beta_0 t)$, which helps mitigate the approximation of the bubble being in an infinite volume of water by adjusting the frequency as it rises and nears the surface. van den Doel (2005) conducted a user study and determined $\xi \approx 0.1$ to be the optimal value for a realistic rise in pitch.

To find the initial amplitude, A_0, in Equation 3.3, (Longuet-Higgins, 1992) considers a bubble with mean radius r_0 that oscillates with a displacement ϵr_0, the pressure p at distance l is given by

$$p(t) = -\frac{4\pi^2 \epsilon r_0^3 f_0^2}{l} sin(2\pi f_0 t). \tag{3.4}$$

Simplifying by plugging in f_0 from Equation (3.1), we see that $|p| \propto \epsilon r_0/l$. Longuet-Higgins plugs in empirically observed values for $|p|$ and suggests that the initial displacement is 1% to 10% of the mean bubble radius r_0. Therefore, we can set

$$A_0 = \epsilon r_0 \tag{3.5}$$

in Equation (3.3), where $\epsilon \in [0.01, 0.1]$ is a tunable parameter that determines the initial excitation of the bubbles. We found that using a power law to select ϵ was effective

$$g(\epsilon) \propto \epsilon^{-\mu}, \tag{3.6}$$

where g is the probability density function of ϵ. By carefully choosing the *scaling exponent* μ, we can ensure that most of the values of ϵ are within the desired range, i.e. below 10%. This gives us a final equation for the pressure wave created by an oscillating spherical bubble (i.e. what travels through the water, then air, to our ear) of

$$p(t) = \epsilon r_0 sin(2\pi f(t) t) e^{-\beta_0 t} \qquad \epsilon \in [0.01, 0.1] \tag{3.7}$$

3.1.2 Generalization to Non-Spherical Bubbles

The approximations given above assume that the shape of the bubble is spherical. Given that an isolated bubble converges to a spherical shape, the previous method is a simple and reasonable approximation. That said, we expect non-spherical bubbles to arise frequently in more complex and

turbulent scenarios. For example, studies of bubble entrapment by ocean waves have shown that breaking waves create long, tube-like bubbles. We illustrate the necessity of handling these types of bubbles in our "dam break" scenario (see Sec. 3.3). Longuet-Higgins also performed a study showing that an initial distortion of the bubble surface of only $\frac{r_0}{2}$ results in a pressure fluctuation as large as $\frac{1}{8}$ atmosphere (Longuet-Higgins, 1989b). Therefore, the shape distortion of bubbles is a very significant mechanism for generating underwater sound. The generated audio also creates a more complete sound, since a single non-spherical bubble will generate multiple frequencies (as can be heard in the accompanying video).

In order to develop a more exact solution for non-spherical bubbles, we consider the deviations from the perfect sphere in the form of spherical harmonics, i.e.

$$r(\theta, \phi) = r_0 + \sum c_n^m Y_n^m(\theta, \phi). \tag{3.8}$$

Section 3.6 of (Leighton, 1994) presents a full derivation for this equation. By solving for the motion of the bubble wall under the influence of the inward pressure, outward pressure and surface tension on the bubble (which depends on the curvature), it can be shown that each zonal spherical harmonic Y_n^0 oscillates at

$$f_n^2 \approx \frac{1}{4\pi^2}(n-1)(n+1)(n+2)\frac{\sigma}{\rho r_0^3} \tag{3.9}$$

where σ is the surface tension. Longuet-Higgins (1992) notes that unlike spherical bubbles, the higher order harmonics decay predominantly due to viscous damping, and not thermal or radiative damping. The amplitude of the n^{th} mode thus decays with $e^{-\beta_n t}$, where

$$\beta_n = (n+2)(2n+1)\frac{\nu}{\rho r_0^2} \tag{3.10}$$

and ν is the kinematic viscosity of the liquid. Given the frequency and damping coefficient for each spherical harmonic, we can again use Equation (3.3) to find the time evolution for each mode. Figure 3.1 gives several examples of oscillation modes corresponding to different spherical harmonics.

Since we have a separate instance of Equation (3.3) for each harmonic mode, we must also determine the amplitude for each mode. The time-varying shape of the bubble can be described by

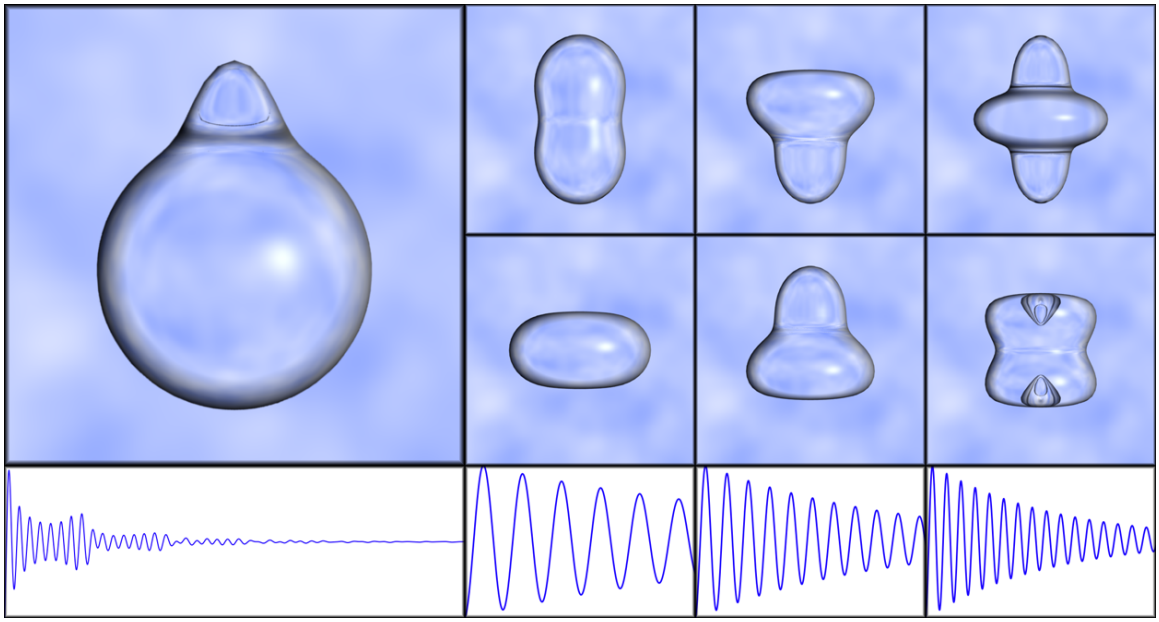


Figure 3.1: Here we show a simple bubble decomposed into spherical harmonics. The upper left shows the original bubble. The two rows on the upper right show the two octaves of the harmonic deviations from the sphere. Along the bottom is the sound generated by the bubble and the components for each harmonic.

the following formula,

$$r(\theta, \varphi; t) \sim r_0 + \sum_n c_n^0(t) Y_n^0(\theta, \varphi) \cos(2\pi f_n t + \vartheta), \tag{3.11}$$

and as with a spherical bubble, each n^{th} harmonic mode radiates a pressure wave p_n as it oscillates. The first-order term of the radiated pressure p_n, when observed at a distance l from the source, depends on $(r_0/l)^{n+1}$ (Longuet-Higgins, 1989b,a), which dies out rapidly and can be safely ignored. The second-order term of the radiated pressure decays as l^{-1} and oscillates at a frequency of $2f_n$, twice as fast as the shape oscillation. Leighton proposes the following equation for p_n

$$\begin{aligned} p_n(t) &= -\frac{1}{l}\left(\frac{(n-1)(n+2)(4n-1)}{2n+1}\frac{\sigma c_n^2}{r_0^2}\right) \\ &\left(\frac{\omega_n^2}{\sqrt{(4\omega_n^2-\omega_b^2)^2+(4\beta_n\omega_n)^2}}\right) e^{-\beta_n t} cos(2\omega_n t) \end{aligned} \tag{3.12}$$

where c_n is the shorthand for c_n^0, the coefficient of the n^{th} zonal spherical harmonic from Equation (3.11), $\omega_n = 2\pi f_n$, $\omega_b = 2\pi f_b = 2\pi(f_0^2 - \beta_0^2)^{\frac{1}{2}}$ is the angular frequency of the radial (0^{th}) mode

(shifted due to damping), and β_n is the damping factor whose value is determined by Equation (3.10). Using Equations (3.10) and (3.12) we can determine the time evolution of each of the n spherical harmonic modes.

In order to determine the number of spherical harmonics to be used, several factors need to be considered. First notice that mode n oscillates at a frequency of $2f_n$, creating a range of n whose resulting pressure waves are audible. We define N_{aud} to be the number of these audible n's. N_{aud} can be derived using Equation (3.9), the radius r_0 of a bubble and the human audible range (20 to 20,000 Hz).

The second term in Equation (3.12) depends on $1/(4\omega_n^2 - \omega_b^2)$, which means that as $2\omega_n$ approaches ω_b (thus $2f_n$ approaches f_b), the n^{th} mode resonates with the 0^{th} mode, and the value of $|p_n|$ increases dramatically, as shown in Figure 3.2. Therefore we select the most important modes in the spherical harmonic decomposition (described in section 3.2.2.4), by choosing values of n with frequencies close to $\frac{1}{2}f_b$ and truncating the rest of the modes (corresponding to the left and the right tails in Figure 3.2). We compute the initial energy for each mode, E_n (proportional to $|p_n|^2$), and collect the modes starting from the largest E_n, until (1) E_n is less than a given percentage, p, of the largest mode, E_{max}; or (2) the sum of energy of the modes not yet selected is less than a percentage, p, of the total energy of all audible modes, E_{total}. The number of modes selected by (1) is denoted as $N_{ind}(p)$, and that by (2) as $N_{tot}(p)$. Some typical values for different r_0's are shown in Table 3.1. One may choose either one of two criteria or a combination of both. As indicated in Table 3.1, 8 modes seems sufficient for various sizes of bubble radii using the criterion (1), where the E_n falls below 1% of E_{max}. Therefore, we can also use a fixed number of modes, say 8 to 10, in practice.

Furthermore, recall that in Equation (3.12) the pressure decays exponentially with a rate β_n, where Equation (3.10) tells us that β_n increases with n and decreases with r_0. If we choose to ignore the initial "burst" and only look at the pressure wave a short time (e.g. 0.001 s) after the creation of the bubble, then we can drop out even more modes at the beginning. This step is optional and the effect is shown in the rightmost two columns of Table 3.1.

Equations (3.7) and (3.12) provide the mechanism for computing the sound generated by either single or multi-mode bubbles, respectively. The pressure waves created by the oscillating bubble travel through the surrounding water, into the air and to the listener. Since we do not consider

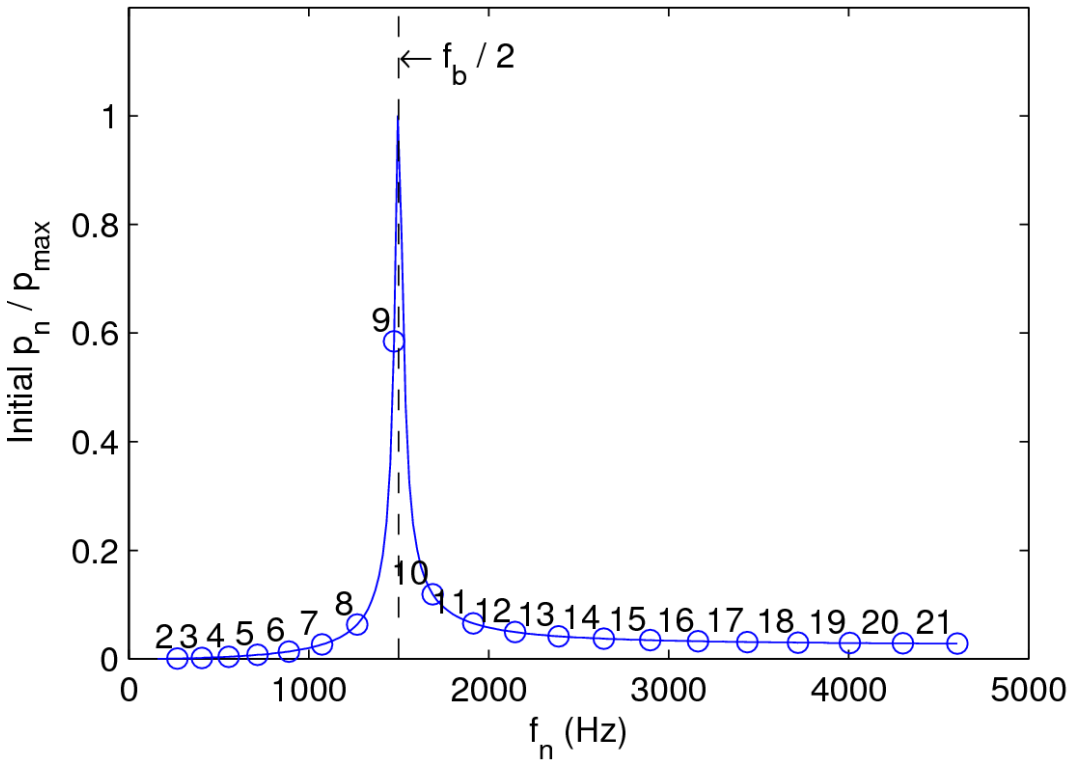


Figure 3.2: A plot of the initial amplitude vs. frequency. From the plot it is clear that as f_n (the frequency of the bubble) approaches $\frac{1}{2}f_b$ (the damping shifted frequency) the initial amplitude increases dramatically. We, therefore, use harmonics where $f_n \approx \frac{1}{2}f_b$ because they have the largest influence on the initial amplitude.

r_0 (m)	N_{aud}	$N_{ind}(1\%)$ $(t = 0)$	$N_{tot}(10\%)$ $(t = 0)$	$N_{ind}(1\%)$ $(t = 10^{-3}\text{s})$	$N_{tot}(10\%)$ $(t = 10^{-3}\text{s})$
0.5	1881	4	1109	4	87
0.05	90	8	106	8	12
0.005	20	4	1	4	1

Table 3.1: **Number of modes selected by the two criteria for various typical r_0's.**

propagation in this chapter, we assume a fixed distance between the listener and each bubble using Equations (3.7) and (3.12) to model the pressure at the listener's ear.

3.1.3 Statistical Generation

In the case where the fluid simulator does not handle bubble generation, we present a statistical approach for generating sound. For a scene at a particular time instant, we consider how many bubbles are created and what they sound like. The former is determined by a bubble generation criteria and the latter is determined by a radius distribution model. As a result, even without knowing the exact motion and interaction of each bubble from the fluid simulator, a statistical approach based on our bubble generation criteria and radius distribution model provide sufficient information for approximating the sound produced in a given scene.

3.1.3.1 Bubble Generation Criteria

Our goal is to examine only the physical and geometrical properties of the simulated fluid, such as fluid velocity and the shape of the fluid surface, and be able to determine when and where a bubble should be generated. Recent works in visual simulation use curvature alone (Narain et al., 2007), or curvature combined with Weber number (Mihalef et al., 2009) as the bubble generation criteria.

In our work, we follow the approach presented by Mihalef et al. (2009). The Weber number is defined as

$$We = \frac{\rho \Delta U^2 L}{(\sigma)} \tag{3.13}$$

where ρ is the density of the fluid, ΔU is the relative gas-liquid velocity, L is the characteristic length of the local liquid geometry and σ is the surface tension coefficient (Sirignano, 2000). This dimensionless number *We* can be viewed as the ratio of the kinetic energy (proportional to $\rho \Delta U^2$) to the surface tension energy (proportional to σ / L). Depending on the local shape, when this ratio is

beyond a critical value, the gas has sufficient kinetic energy to "break into" the liquid surface and form a bubble; while at lower Weber numbers, the surface tension energy is able to separate the water and air.

Besides the Weber number, we also need to consider the limitation of a fluid simulator. In computer graphics, fluid dynamics is usually solved on a large-scale grid, with small-scale details such as bubbles and droplets added in at regions where the large-scale simulation behaves poorly, namely regions of high curvature. This is because a bubble is formed when the water surface curls back and closes up, at which site the local curvature is high.

Combining the effects of the Weber number and the local geometry, we evaluate the following parameter on the fluid surface

$$\Gamma = u^2\kappa, \tag{3.14}$$

where u is the liquid velocity and κ is the local curvature of the surface. The term u^2 encodes the Weber number, because in Equation 3.13 ρ, σ and L (which is taken to be the simulation grid length dx) are constants, and $\Delta U^2 = u^2$ since the air is assumed to be static. Bubbles are generated at regions where Γ is greater than a threshold Γ_0. The criteria also matches what we observe in nature–a rapid river (larger u) is more likely to trap bubbles than a slow one. In the ocean, bubbles are more likely to form near a wave (larger κ) than on a flat surface–our bubble generation mechanism captures both of these characteristics.

3.1.3.2 Bubble Distribution Model

Once we have determined a location for a new bubble using the generation criteria, we select its radius at random according to a radius distribution model. Works on bubble entrapment by rain (Pumphrey and Elmore, 1990) and ocean waves (Deane and Stokes, 2002) suggest that bubbles are created in a power law ($r^{-\alpha}$) distribution, where α determines the ratio of small to large bubbles. In nature, the α takes value from 1.5 to 3.3 for breaking ocean waves (Deane and Stokes, 2002) and ≈ 2.9 for rain (Pumphrey and Elmore, 1990), thus in simulation it can be set according to the scenario. The radius affects both the oscillation frequency (Equation 3.1) and the initial excitation (Equation 3.5) of the bubble. Plugging in the initial excitation factor ϵ selected by Equation 3.6, the sound for the bubble can be fully determined by Equation 3.7. Combining the genration criteria

and the radius distribution model, our approach approximate the number of sound sources and the characteristics of their sounds plausibly in a physically-based manner for a dynamic scene.

3.2 Integration with Fluid Dynamics

There are many challenging computational issues in the direct coupling of fluid simulation with sound synthesis. As mentioned earlier, the three commonly used categories of fluid dynamics in visual simulation are grid-based methods, SPH and shallow-water approximations. We consider two fluid simulators that utilize all three of these methods. Our shallow water formulation is an integrated adaptation of the work of Thürey et al. (2007a; 2007b) and Hess (2007). The other is a hybrid grid-SPH approach, taken heavily from the work of Hong et al. (2008). We present a brief overview of the fluid simulator methods below and describe how we augment the existing fluid simulation methods to generate audio. We refer the reader to (Thürey et al., 2007a; Hess, 2007; Hong et al., 2008) for full details on the fluid dynamics simulations.

3.2.1 Shallow Water Method

3.2.1.1 Dynamics Equations

The shallow water equations approximate the full Navier-Stokes equations by reducing the dimensionality from 3D to 2D, with the water surface represented as a height field. This approximation works well for situations where the velocity of the fluid does not vary along the vertical axis and the liquid has low viscosity. The height field approximation restricts us to a single value for the fluid along the vertical axis, making it unable to model breaking waves or other similar phenomena.

The evolution of the height field, $H(x,t)$, in time is governed by the following equations:

$$\frac{\partial H}{\partial t} = -v \cdot \nabla H - H(\frac{\partial v_x}{\partial x} + \frac{\partial v_y}{\partial y})$$

$$\frac{\partial v_x}{\partial t} = -v \cdot \nabla v_x - g\frac{\partial H}{\partial x}$$

$$\frac{\partial v_y}{\partial t} = -v \cdot \nabla v_y - g\frac{\partial H}{\partial y}$$

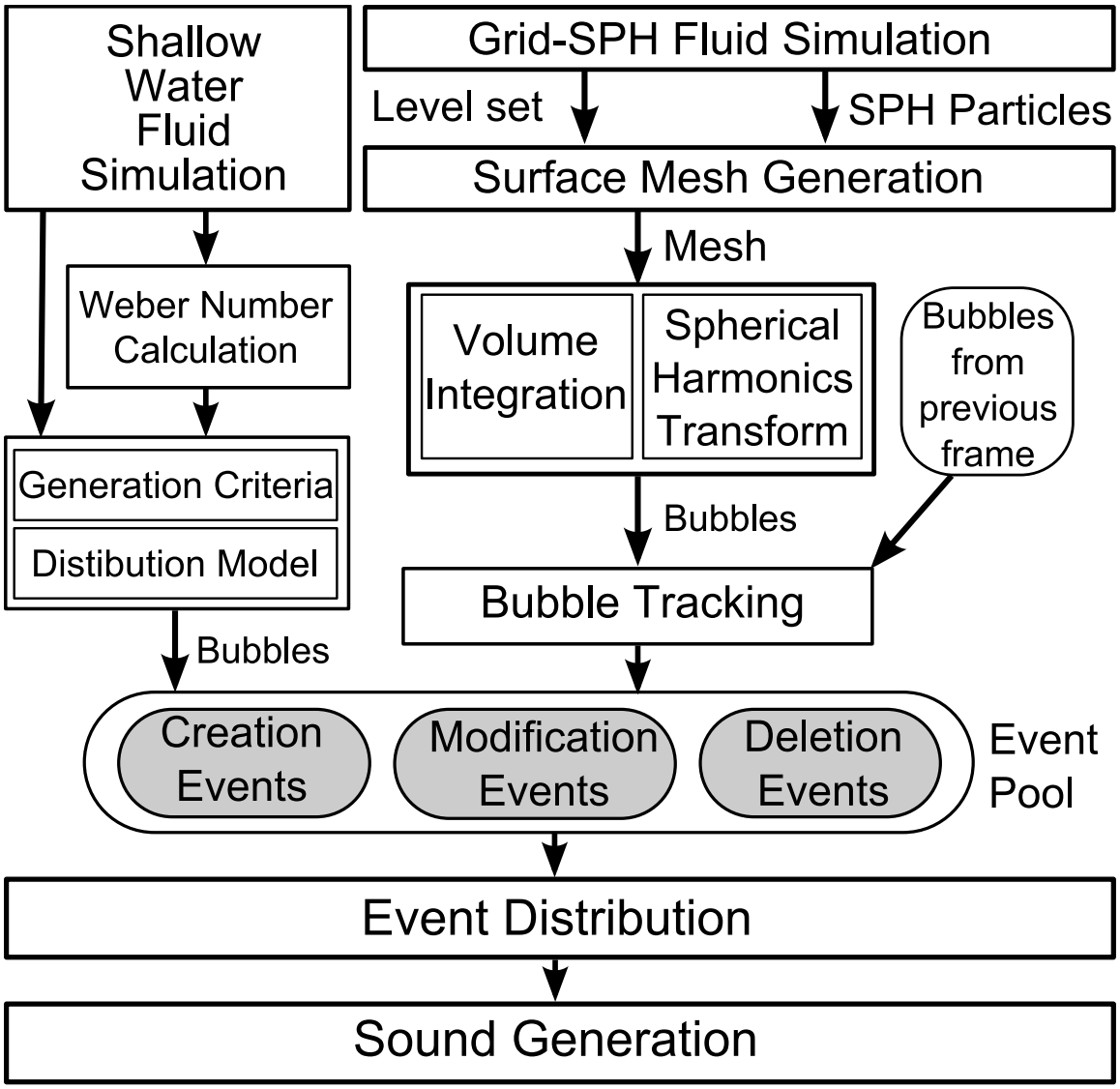


Figure 3.3: An overview of our liquid sound synthesis system

where we assume the gravitation force, $g = (0, 0, g)^T$ is along the z-axis and v is the horizontal velocity of the fluid. We use a staggered grid of size $N_x \times N_y$ with equal grid spacing Δx and use a semi-Lagrangian advection step to solve the equations.

3.2.1.2 Rigid Bodies

Due to the 2D nature of the shallow water equations, rigid bodies must be explicitly modeled and coupled to the fluid simulation. This is complicated by the fact that our rigid bodies are 3D, whereas, our fluid simulation is 2D. We therefore cannot apply the method for fluid-rigid body coupling presented in previous works (Carlson et al., 2004; Batty et al., 2007; Robinson-Mosher et al., 2008), as our cells encompass an entire column of water and it is unlikely a rigid body will be large enough to fill a full vertical column. To that end, we explicitly model the interactions between the fluid simulation and the rigid body simulation using two one-way coupling steps.

The rigid body is coupled to the fluid in two ways, a buoyancy force and drag and lift forces resulting from the fluid velocity. The buoyancy force is calculated by projecting the area of each triangle up to the water surface, counting downward facing triangles positive and upward facing ones negative. The resulting force is calculated as,

$$f_{bouy} = -g\rho \sum_{i=1}^{n} -sign(n_i \cdot e_z)V_i,$$

where ρ is the density of the fluid, n_i and V_i are the normal and projected volume of triangle i and e_z points in the upward direction. The drag and lift forces are also calculated per face and point opposite and tangential to the relative velocity of the face and the fluid, respectively. Exact equations can be found in (Hess, 2007).

The fluid is coupled to the object in two ways as well, through the surface height and the fluid velocity. The height is adjusted based on the amount of water displaced by the body on a given time step. This is again calculated per face, but this time the face is projected in the direction of the relative velocity. This can create both positive and negative values for the volume displaced, which is desirable for generating both the wave in front of a moving body and the wake behind. The fluid velocity of the cells surrounding a rigid body are adjusted as the water is dragged along with the

body. The adjustment is calculated used the percentage of the column of water filled by the rigid body, the relative velocities and a scaling constant. More details can again be found in (Hess, 2007).

3.2.2 Grid-SPH Hybrid Method

3.2.2.1 Dynamics Equations

We use an octree grid to solve the invicid incompressible Navier-Stokes equations (Losasso et al., 2004), which are

$$u_f + (u \cdot \nabla)u + \nabla p/\rho = f$$

$$\nabla \cdot u = 0$$

where u is the fluid velocity, p is the pressure, ρ is the density and f is the external forcing term. Although this provides a highly detailed simulation of the water, it would be too computationally expensive to refine the grid down to the level required to simulate the smallest bubbles. To resolve this, we couple the grid-based solver with bubble particles, modeled using SPH particles (Müller et al., 2003, 2005; Adams et al., 2007). The motion of the particles is determined by the sum of the forces acting on that particle. The density of particles at a point i defined as $\rho_i = \sum m_j W(x_{ij}, r_j)$ where $W(x, r)$ is the radial symmetric basis function with support r defined in (Müller et al., 2003) and m_j and r_j are the mass and radius of particle j. We therefore model the interactions of the bubbles with the fluid simulator through a series of forces acting on the bubble particles:

1. A repulsive force to model the pressure between air particles, that drops to zero outside the support $W(x, r)$

2. Drag and lift forces defined in terms of the velocity at the grid cells and the radius and volume of the particles, respectively

3. A heuristic vorticity confinement term based on the vorticity confinement term from (Fedkiw et al., 2001)

4. A cohesive force between bubble particles to model the high contrast between the densities of the surrounding water and the air particles

5. A buoyancy force proportional to the volume of the particle

To model the effects of the bubbles on the water, we add the reactionary forces from the drag and lift forces mentioned above as external forcing terms into the incompressible Navier-Stokes equations given above.

3.2.2.2 Bubble Extraction

Specifically, we need to handle two types of bubbles, those formed by the level sets and those formed by the SPH particles. The level set bubbles can be separated from the rest of the mesh returned by the level set method because they lie completely beneath the water surface and form fully connected components. Once we have meshes representing the surface of the bubbles, we decompose each mesh into spherical harmonics that approximate the shape, using the algorithm presented in Section 3.2.2.4. The spherical harmonic decomposition and the subsequent sound synthesis is linear in the number of harmonic modes calculated. Therefore, the number of spherical harmonics calculated can be adjusted depending on desired accuracy and available computation time (as discussed in Sec. 3.1.2). Once we have the desired number of spherical harmonics, we determine the resonant frequencies using Equation (3.9).

For SPH bubble particles, there are two cases–when a bubble is represented by a single particle and when it is represented by multiple particles. In the case of a single particle bubble we simply use the radius and Equation (3.7) to generate the sound. When multiple SPH particles form one bubble, we need to determine the surface formed by the bubble. We first cluster the particles into groups that form a single bubble and then use the classic marching cubes algorithm (Lorensen and Cline, 1987) within each cluster to compute the surface of the bubble. Once we have the surface of the bubble, we use the same method as the level set bubble to find the spherical harmonics and generate audio.

3.2.2.3 Bubble Tracking and Merging

At each time step the fluid simulator returns a list of level set bubble meshes and SPH particles which we convert into a set of meshes, each representing a single bubble. At each subsequent time step we collect a new set of meshes and compare it to the set of meshes from the previous time step with the goal of identifying which bubbles are new, which are preexisting and which have disappeared. For each mesh, M, we attempt to pair it with another mesh, M_{prev}, from the previous time step such that they represent the same bubble after moving and deforming within the time step.

We first choose a distance, $l \geq v_{max}\Delta t$, where v_{max} is the maximum possible speed of a bubble. We then define `neighbor`(M, l) as the set of meshes from the previous time step whose center of masses lie within l of M. For each mesh in `neighbor`(M, l), we compute its *similarity score* based on the proximity of its center of mass to M and the closeness of the two volumes, choosing the mesh with the highest similarity score. Once we have created all possible pairs of meshes between the new and the old time steps, we are left with a set of bubbles from the old time step with no pair–the bubbles to remove–and a set of bubbles in the new time step–the bubbles to create. Although it may be possible to create slightly more accurate algorithm by tracking the particles that define an SPH or level set bubble, these methods would also present nontrivial challenges. For example, in the case of tracking the level set bubbles, the level set particles are not guaranteed to be spaced in any particular manner and are constantly added and deleted, making this information difficult to use. In the case of tracking bubbles formed by SPH particles, there would still be issues related to bubbles formed by multiple SPH particles. The shape could remain primarily unchanged with the addition or removal of a single particle and therefore the audio should remain unchanged as well, even though the IDs of the particles change. We chose this approach because of its generality and its ability to handle both level set and SPH bubbles, as well as other types of fluid simulators.

3.2.2.4 Spherical Harmonic Decomposition

In order to decompose a mesh, M, into a set of the spherical harmonics that approximate it, we assume that M is a closed triangulated surface mesh and that it is *star-shaped*. A mesh is *star-shaped* if there is a point o such that for every point p on the surface of M, segment $\overline{op}$ lies entirely within M. The length of the segment $\overline{op}$ can be described as a function $|\overline{op}| = r(\theta, \varphi)$ where θ and φ are the polar and azimuthal angles of p in a spherical coordinate system originating at o. The function $r(\theta, \varphi)$ can be expanded as a linear combination of spherical harmonic functions as in Equation (3.8).

The coefficient c_n^m can be computed through an inverse transform

$$c_n^m = \int_\Omega P(\theta, \varphi)\overline{Y}_n^m(\theta, \varphi)d\Omega$$

where the integration is taken over Ω, the solid angle corresponding to the entire space. Furthermore, if T is a triangle in M and we define the solid angle spanned by T as Ω_T, then we have $\Omega = \bigcup_{T \in M} \Omega_T$

and $c_n^m = \sum_{T \in M} \int_{\Omega_T} P(\theta, \varphi) \overline{Y}_n^m(\theta, \varphi) d\Omega$. The integration can be calculated numerically by sampling the integrand at a number of points on each triangle. For sound generation, we only need the zonal coefficients c_n^0, with n up to a user defined bandwidth, B. The spherical harmonic transform runs in $O(BN_p)$ where N_p is the total number of sampled points.

If the bubble mesh is not star-shaped, then it cannot be decomposed into spherical harmonics using Equation (3.8). To ensure that we generate sound for all scenarios, if our algorithm cannot find a spherical harmonic decomposition it automatically switches to a single mode approximation based on the total volume of the bubble. Since this only happens with large, low-frequency bubbles, we have not noticed any significant issues resulting from this approximation or the transition between the two generation methods.

3.2.3 Decoupling Sound Update from Graphical Rendering

Since computing the fluid dynamics at 44,000 Hz, the standard frequency for good quality audio, would add an enormous computation burden, we need to reconcile the difference between the fluid simulator time step, T_{sim} (30-60 Hz), and the audio generation time step, T_{audio} (44,000 Hz). We can use Equations (3.1) and (3.9) to calculate the resonant frequency at each T_{sim} and then use Equations (3.7) and (3.12) to generate the impulse response for all the T_{audio}'s until the subsequent T_{sim}. Naively computing the impulse response at each T_{audio} can create complications due to a large number of events that take place in phase at each T_{sim}. In order to resolve this problem, we randomly distribute each creation, merge and deletion event from T_{sim} onto one of the ~733 T_{audio} between the current and last T_{sim}.

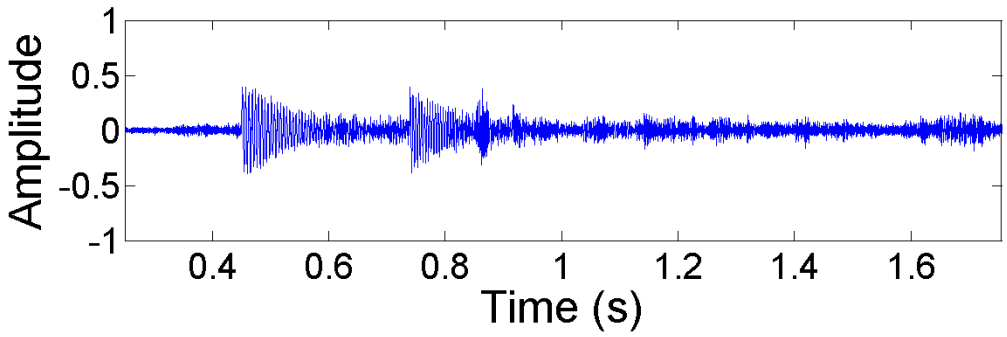
3.3 Implementation and Results

The rendering for the shallow water simulation is performed in real time using OpenGL and custom vertex and fragment shaders while the rendering for the hybrid simulator is done off-line using a forward ray tracer. In both cases, once the amplitude and frequency of the bubble sound is calculated, the final audio is rendered using The Synthesis ToolKit (Cook and Scavone, 2010).

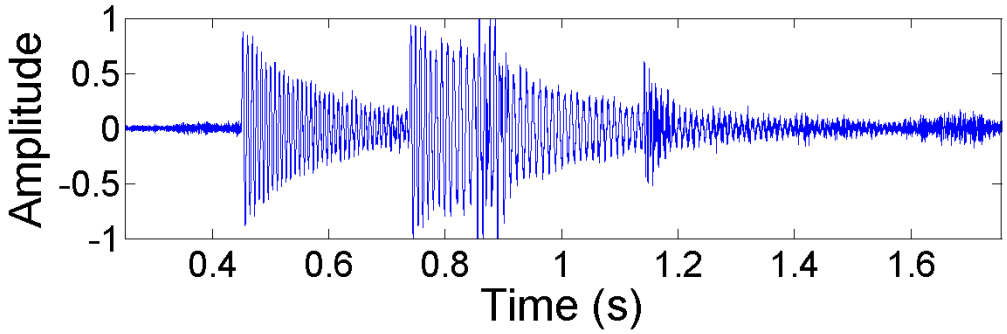
3.3.1 Benchmarks

We have tested our integrated sound synthesis system on the following scenarios (as shown in the supplementary videos).

3.3.1.1 Hybrid Grid-SPH Simulator



(a) Spherical Harmonic Decomposition



(b) Minimum Enclosing Sphere

Figure 3.4: Wave plots showing the frequency response of the pouring benchmark. We have highlighted the moments surrounding the initial impact of the water and show our method (top) and a single-mode method (bottom) where the frequency for each bubble is calculated using volume of the minimum enclosing sphere.

Pouring Water: In this scenario, water is poured from a spigot above the surface as shown in Figure 3.5. The initial impact creates a large bubble as well as many smaller bubbles. The large bubble disperses into smaller bubbles as it is bombarded with water from above. The generated sound takes into account the larger bubbles as well as all the smaller ones, generating the broad spectrum of sound heard in the supplementary video. An average of 11,634 bubbles were processed per simulation frame to generate the sounds. Figure 3.4 shows plots of the sound generated using our

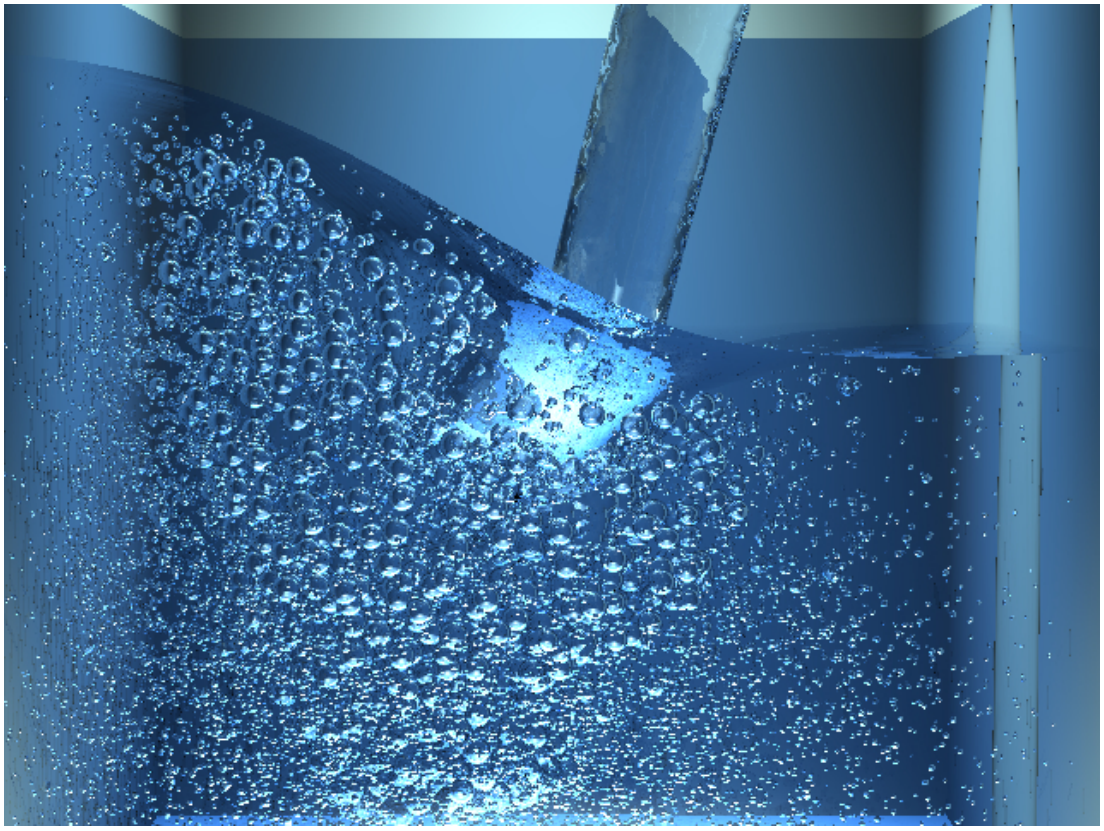
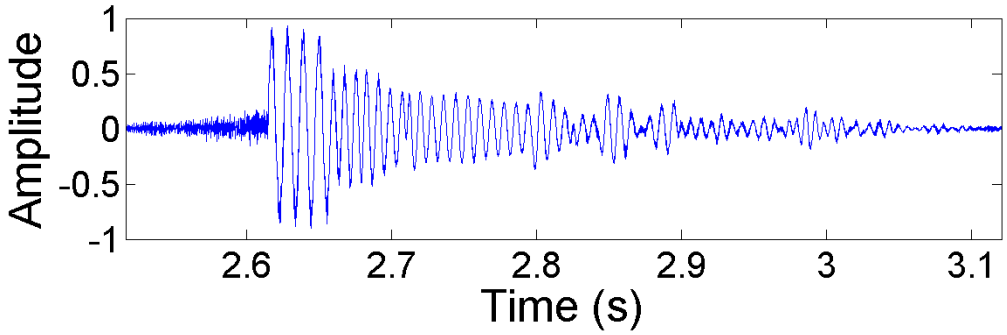


Figure 3.5: Liquid sounds are generated automatically from a visual simulation of pouring water.

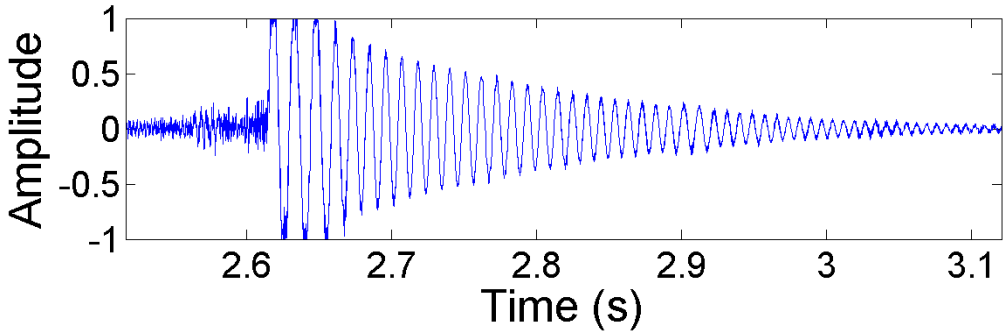
method and a single-mode version using the volume of the minimum enclosing sphere to calculate the volume.

Five Objects: In this benchmark, shown in Figure 3.7, five objects are dropped into a tank of water in rapid succession, creating many small bubbles and one large bubble as each one plunges beneath the water surface. The video shows the animation and the sound resulting from the initial impacts as well as the subsequent bubbles and sound generated by the sloshing of the water around the tank. We used ten spherical harmonic modes and processed up to 15,000 bubbles in a single frame. Figure 3.6 shows the wave plots for our method and the minimum enclosing sphere method. As you can see, using the spherical harmonic decomposition creates a fuller sound, whereas the minimum enclosing sphere method creates one frequency that decays over time.

Dam Break: In this benchmark, shown in Figure 3.9, we simulate the ”dam break” scenario that has been used before in fluid simulation, however, we generate the associated audio automatically. We processed an average of 13,589 bubbles per frame using five spherical harmonic modes. This



(a) Spherical Harmonic Decomposition



(b) Minimum Enclosing Sphere

Figure 3.6: Wave plots showing the frequency response of the five objects benchmark. We have highlighted the impact of the final, largest object. The top plot shows our method and the bottom, a single-mode method where the frequency for each bubble is calculated using volume of the minimum enclosing sphere.

benchmark also demonstrates the creation of a tube-shaped bubble as the right-to-left wave breaks, something that studies in engineering (Longuet-Higgins, 1990) have shown to be the expected result of wave breaking. The creation of highly non-spherical, tube-like bubbles highlight the need for the spherical harmonic decomposition to handle bubbles of arbitrary shapes. This is illustrated in the supplementary video and Figure 3.8, where the minimum enclosing sphere method creates a highly distorted wave plot when the tube-shaped bubble is created.

3.3.1.2 Shallow Water Simulator

Brook: Here we simulate the sound of water as it flows in a small brook. We demonstrate the interactive nature of our method by increasing the flow of water half way through the demo, resulting in higher velocities and curvatures of the water surface and therefore, louder and more turbulent sound.

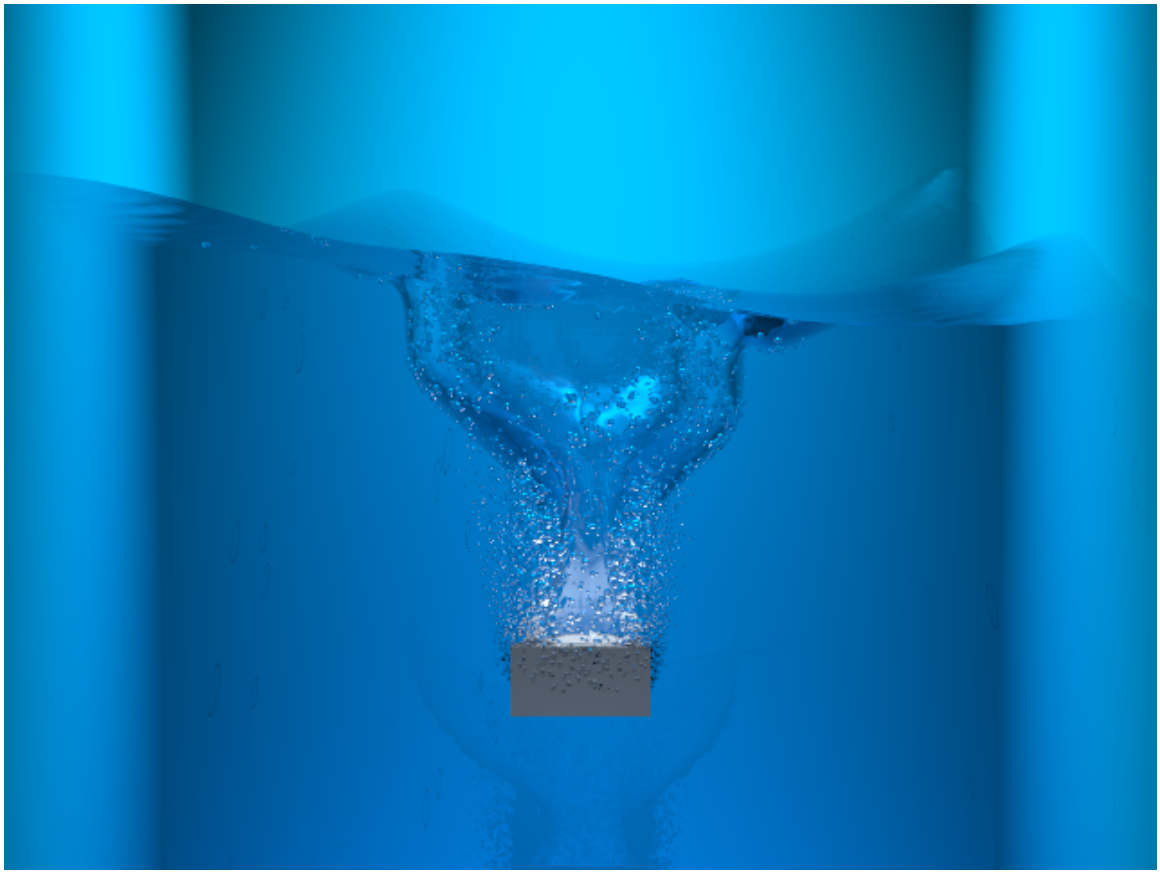
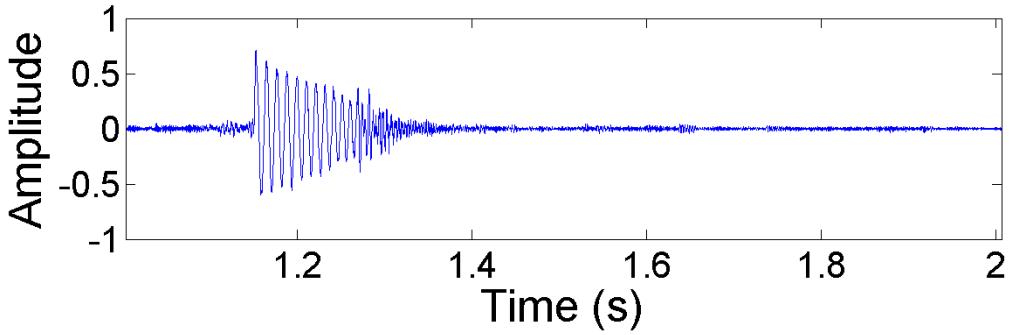


Figure 3.7: Sound is generated as five objects fall into a tank of water one after another.

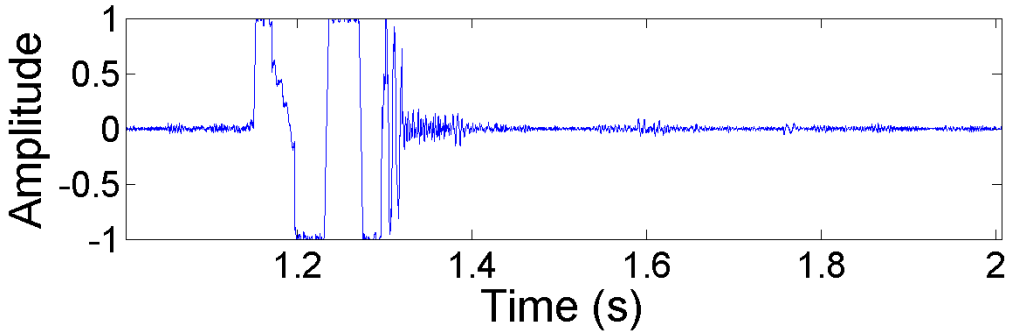
Duck: As shown in Figure 3.11, as a user interactively moves a duck around a bathtub, our algorithm automatically generates the associated audio. The waves created by the duck produces regions of high curvature and velocity, creating resonating bubbles.

3.3.2 Timings

Tables 3.2 and 3.3 show the timings for our system running on a single core of a 2.66GHz Intel Xeon X5355. Table 3.2 shows the number of seconds per frame for our sound synthesis method integrated with grid-SPH hybrid method. Column two displays the compute time of the fluid simulator (Hong et al., 2008). Columns three, four and five break down the specifics of the synthesis process, and column six provides the total synthesis time. Column three represents the time spent extracting the bubble surface meshes from the level set and SPH particles (described in section 3.2.2.2). Column four is the time spent performing the spherical harmonic decomposition and



(a) Spherical Harmonic Decomposition



(b) Minimum Enclosing Sphere

Figure 3.8: Wave plots showing the frequency response for the dam break scenario. We highlight the moment when the second wave crashes (from right to left) forming a tube-shaped bubble. The top plot shows our method and the bottom, a single-mode method where the frequency for each bubble is calculated using volume of the minimum enclosing sphere.

spherical volume calculation (section 3.1.2) and column five is the time spent tracking the bubbles (section 3.2.2.3) and generating the audio (section 3.1).

	Average Bubbles per Frame	Fluid Simulation	Sound Synthesis			
			Surface Generation	Bubble Integration	Tracking & Rendering	Total
Pouring	11,634	1,259 s	10.20 s	1.77 s	0.18 s	12.15 s
Five Objects	1,709	1,119 s	2.37 s	0.21 s	0.94 s	3.52 s
Dam Break	13,987	3,460 s	39.92 s	1.45 s	1.13 s	42.50 s

Table 3.2: **Hybrid Grid-SPH Benchmark Timings (seconds per frame).**

Table 3.3 show the timings the shallow water simulator. Column one (Simulation) includes the time for both the shallow water simulation and the sound synthesis and column two (Display) is the time required to graphically render the water surface and scene to the screen. From the table we

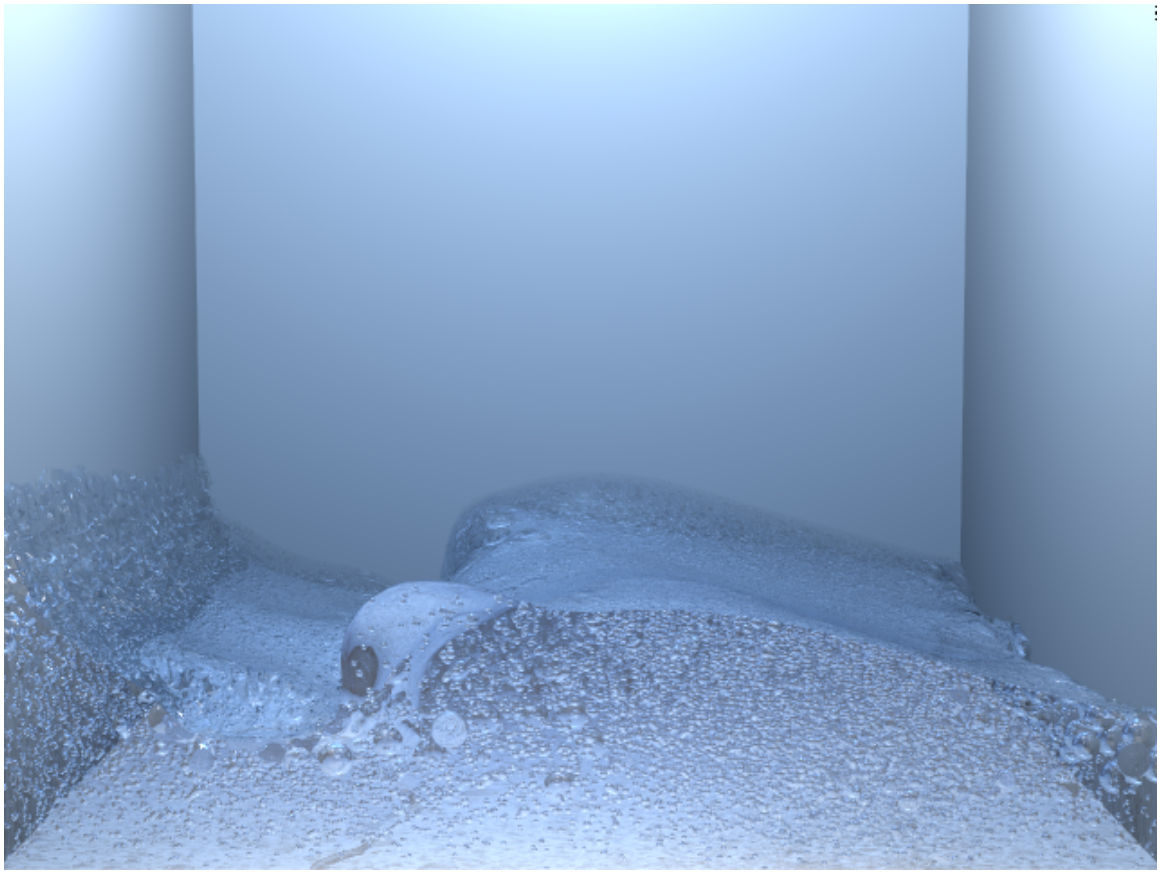


Figure 3.9: A “dam-break” scenario, a wall of water is released, creating turbulent waves and sound as the water reflects off the far wall.

can see that both simulations run at around 55 frames per second, leaving compute time for other functions while remaining real-time.

Figure 3.10: Real-time sounds are automatically generated from an interactive simulation of a creek flowing through a meadow.

	Simulation	Display
Creek Flowing	4.74 msec	12.80 msec
Duck in the Tub	7.59 msec	10.93 msec

Table 3.3: **Shallow Water Benchmark Timings (msec per frame).**

3.3.3 Comparison with Harmonic Fluids

A quick comparison of the timings for our method vs. Harmonic Fluids shows that our shallow water sound synthesis technique runs in real time, including sound synthesis, fluid simulation, and graphical rendering. This makes our approach highly suitable for many real-time applications, like virtual environments or computer games. It is also important to note that our benchmarks highlight more turbulent scenarios than those shown in (Zheng and James, 2009), thus generating more bubbles *per simulation frame*. Our method also runs in a few seconds on a typical single-core PC, instead

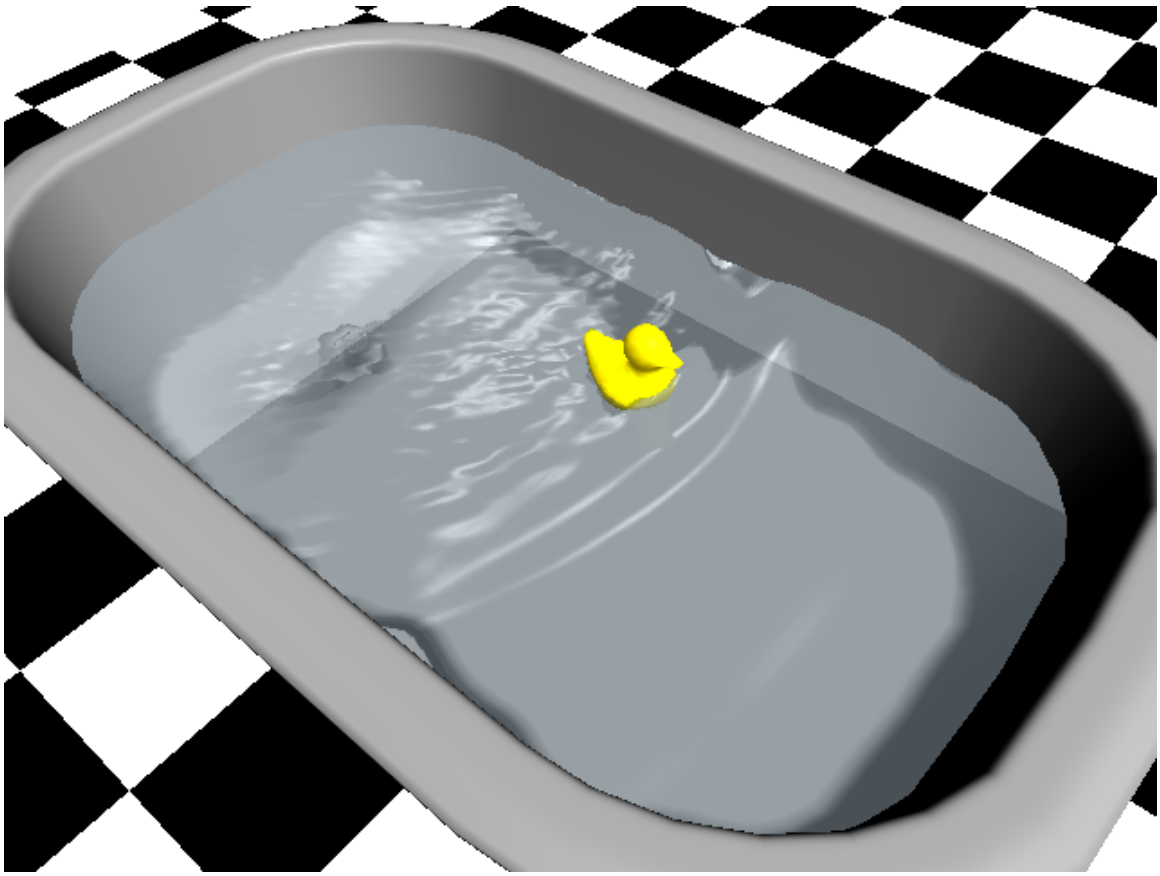


Figure 3.11: Sounds are automatically generated as a (invisible) user moves a duck in a bathtub.

of many hours on a many-core platform (such as (Zheng and James, 2009) for computing sound radiation). The most time-consuming step in our current implementation is surface extraction using a standard Marching Cubes algorithm (Lorensen and Cline, 1987). A more efficient variation of the Marching Cubes algorithm could offer additional performance improvements.

3.4 User Study

To assess the effectiveness of our approach, we designed a set of experiments to solicit user feedback on our method. Specifically, we were looking to explore (a) the perceived realism of our method relative to real audio, video without audio, and video with less than perfectly synched audio and (b) whether subjects can determine a difference and have a preference between our method and a simple approximation based on a single-mode bubble. The study consists of four parts, each

containing a series of audio or video clips. The next section details the procedure for each section of our user study.

3.4.1 Procedure

In sections I and II, each subject is presented with a series of audio or video clips. In both cases, one clip is shown per page and the subject is asked to rate the clip on a scale from 1 to 10, with 1 labeled "Not Realistic" and 10 labeled "Very Realistic." In sections III and IV, the subject is shown two audio or video clips side by side. In both cases, the subject is asked "Are these two audio/video clips the same or different?" If they respond "different", we then ask "Which audio/video clip do you prefer?" and "How strongly do you feel about this preference?" The following sections detail the specific video and audio clips shown. In all the sections, the order of the clips is randomized and in sections III and IV, which clip appears on the left or the right is also random. The subject is also always given the option to skip either an individual question or an entire section and can, of course, quit at any time.

Section I: In this section the subject is shown a series of audio clips. The clips consist of five audio clips from our method and four real audio recordings of natural phenomena.

Section II: In this section, the subject is shown a series of video clips. These videos consist of the five benchmarks we produced, each shown with and without the audio we generated.

Section III: Here the subject is presented with six pairs of audio clips. Each page contains the audio from one of our demo scenarios generated using the hybrid grid-SPH simulator paired with either the identical audio clip (to establish a baseline) or the same demo scenario using audio generated with the simplified, Minimal Enclosing Sphere method (denoted as MES in the table).

Section IV: This section is very similar to the previous experimental setup, however, we show the subjects the video associated with the audio they just heard. There are nine pairs of videos. Each page again contains the video and audio from one of our demo scenarios generated using the hybrid grid-SPH simulator paired with either the identical video clip (again, to establish a baseline), the video clip using the Minimal Enclosing Sphere Method or a video clip where we acted as the foley artist, mixing and syncing pre-existing audio clips to our video clip. By adding the video clip with pre-existing audio clips, we intended to evaluate the experience of using manually synched pre-recorded audio clips compared to the audio-visual experience of using our method.

3.4.2 Results

	Mean	Std.	Mean Diff.	Std.
Beach	7.45	2.14	1.67	1.92
Raining	8.69	1.57	2.9	1.53
River	8.17	1.79	2.37	1.57
Splash	7.04	2.44	1.25	2
Pouring	4.74	2.33	-1.05	1.73
Five Objects	4.73	2.26	-1.07	1.52
Dam Break	4.92	2.17	-0.87	1.56
Brook	5.23	2.25	-0.56	1.88
Duck	6.69	2.18	0.89	1.75

Table 3.4: **Section I Results: Audio Only.** The means and standard deviations for section I. Column one is the mean score given by the subject, whereas, column three is the mean of the difference a given question's score was from the mean score for this subject. We calculated this quantity in attempt to mitigate the problem of some subjects scoring all clips high and some subjects scoring all clips low. The top group represents the real sounds and the bottom group represents the sounds generated using our method. All 97 subjects participated in this section.

	Mean	Std.	Mean Diff.	Std.
Pouring	5.95	2.16	0.3	1.66
Pouring (No audio)	4.91	2.22	-0.65	1.7
Five Objects	6.65	2.18	1	1.57
Five Objects (No audio)	6.02	2.48	0.41	1.86
Dam Break	5.87	2.3	0.22	1.72
Dam Break (No audio)	5.36	2.48	-0.23	1.85
Brook	4.52	2.49	-1.13	1.84
Brook (No audio)	3.83	2.29	-1.78	1.61
Duck	6.3	2.45	0.65	2.23
Duck (No audio)	4.92	2.33	-0.7	2.01

Table 3.5: **Section II Results: Video vs. Visual Only.** The means and standard deviations for section II. Column one is the mean score given by the subjects, whereas column three is the mean of the difference a given question's score was from the mean score for this subject. A total of 87 out of 97 subjects chose to participated in this section.

Tables 3.4, 3.5, 3.6 and 3.7 show the results from Sections I - IV of our user study. In many of the subsequent sections we refer to the difference of means test. The test looks at the means and standard errors of two groups of subjects, and determines whether or not we can reject the null hypothesis that the difference we observe between the two means is the result of chance or is statistically significant. The formula for the difference of means can be found in most introductory statistics texts, but we

	Same	Diff	Prefer Ours	Prefer MES	Mean Strength Ours	Mean Strength MES
Pouring	21.8% (17)	78.2% (61)	68.9% (42)	31.1% (19)	6.36	5.42
Five Objects	27.6% (21)	72.4% (55)	54.7% (29)	45.3% (24)	5.86	5.17
Dam Break	2.6% (2)	97.4% (76)	77.3% (58)	22.7% (17)	7.29	5.82

Table 3.6: **Section III Results: Audio Only for Ours vs. Single-Mode.** Columns one and two show the percentage (and absolute number) of people who found our videos to be the same or different than the minimal enclosing sphere method. Columns three and four show, of the people who said they were different, the percentage that preferred ours or the MES method and finally columns five and six show the mean of the stated strength of the preference for those who preferred our method and the MES method. A total of 78 subjects participated in this section.

	Same	Diff	Prefer Ours	Prefer Other	Mean Strength Ours	Mean Strength Other
Pouring	16.7% (12)	83.3% (60)	73.3% (44)	26.7% (16)	6.75	5.75
Five Objects	43.2% (32)	56.8% (42)	48.7% (19)	51.3% (20)	6.42	6.2
Dam Break	5.3% (4)	94.7% (71)	83.3% (55)	16.7% (11)	7.35	6.64
Pouring	1.4% (1)	98.6% (72)	65.7% (46)	34.3% (24)	7.13	6.79
Five Objects	1.3% (1)	98.7% (74)	94.4% (67)	5.6% (4)	8.75	5.33
Dam Break	2.8% (2)	97.2% (69)	60.6% (40)	39.4% (26)	7.65	7.19

Table 3.7: **Section IV Results: Video for Ours vs. Single-Mode(top) & Ours vs. Recorded(bottom).** The top group shows our method versus the minimal enclosing sphere method and the bottom group shows our method versus the prerecorded and synched sounds. Columns one and two show the percentage (and absolute number) of people who found the two videos to be the same or different. Columns three and four show, of the people who said they were different, the percentage that preferred ours or the other method (either MES or prerecorded) and finally columns five and six show the mean of the stated strength of the preference for those who preferred our method and the other method. A total of 75 subjects participated in this section.

present it below for reference:

$$t = \frac{\Delta M_{observed} - \Delta M_{expected}}{\sqrt{SE_1^2 + SE_2^2}}$$

where $\Delta M_{observed}$ is the difference of the observed means, $\Delta M_{expected}$ is the expected difference of the means (for the null hypothesis, this is always 0) and SE_1 and SE_2 are the standard errors for the two observed means (where $SE = \sigma / \sqrt{N}$). t is the t-value of that difference of means test and we choose a value of three on that t-distribution as our cutoff to determine if the difference between the two means is statically significant.

3.4.2.1 Demographics

A total of 97 subjects participated in our study and they were allowed to quit during any section, at any time. 72% of our subjects where male and 28% were female. Their ages ranged from 17 to 65, with a mean of 25. About 82% of subjects owned an iPod or other portable music device and listened to an average of 13 hours of music per week.

3.4.2.2 Mean Subject Difference

Tables 3.4 and 3.5 show the two sections where the subject was asked to rate each video or audio clip individually. For those two sections, along with calculating a regular mean and standard deviation, we also computed a measure that we call the "mean subject difference". Some subjects tended to rate everything low, while some tended to rate everything high. Such individual bias could unnecessarily increase the standard deviation–especially since these ratings are most valuable when compared to other questions in each section. To calculate the mean subject difference, we first take the mean across all questions in a section for each subject, then instead of examining the absolute score for any given question we examine the difference from the mean. So, the mean values will be centered around 0, with the ones subjects preferred as positive.

3.4.2.3 Section I and II

Tables 3.4 and 3.5 present a few interesting results. As we noted above, the subjects were allowed to skip any question or any section of the study. While 97 people participated in section I, only 87 participated in section II. In Table 3.4, the difference of means test clearly shows that the difference between the mean of the real sounds and the computer synthesized sounds is statistically significant. This difference is not surprising given the extra auditory clues that recorded sounds have that synthesized sounds lack. That said, the mean for the duck being moved interactively in the bathtub and the real splashing sound are not statistically different. In the best case, our method is able to produce sounds with comparable perceived realism to recorded sounds. In addition, in three recorded sounds (beach, raining and river), there are multiple sound cues from nature, such as wind, birds and acoustic effects of the space where the recordings were taken. We conjecture that the subjects tend to rate them higher because of the multiple aural cues that strengthen the overall

experience. Therefore, although the perceived realism of our synthesized sounds is scored lower than the perceived realism of the recorded sounds, the fact that our synthesized sounds are no more than one standard deviation away from the recorded sounds without the presence of multiple aural cues is notable.

In Table 3.5, two benchmarks have a statistically significant difference between the means of the video with and without audio: the duck in the bathtub and the pouring water demos. It shows that for these two cases, we can conclusively state that the sound effects generated using our method enhances the perceived realism for the subjects. Although the the results of other cases are statistically inconclusive, they show a difference in the means that suggests the perceived realism is enhanced by using audio generated using our methods.

When comparing the perceived realism of audio only, visual only, and visual with audio from Tables 3.4 and 3.5, we see that for demos with less realistic graphics, like the flowing creek and the duck in the tub, the combined visual-audio experience does not surpass the perceived realism of the audio alone. For benchmarks with more realistic rendering, this is not the case, suggesting that the subject's perception of realism is heavily influenced by the visual cues, as well as the audio.

3.4.2.4 Our method vs. Single-Mode Approximation

Based on the results from Tables 3.6 and 3.7, subjects clearly preferred our method to the method using the minimal enclosing sphere approximation. We believe these studies suggest that when presented with a clear choice, the subjects prefer our method. In addition, the degree of preference, as indicated by the "mean strength" for our method is more pronounced. We also see that the percentage of people who were able to discern the difference between the sounds generated by our method vs. MES approximation is highest in the Dam-Break benchmark, where the bubbles were most non-spherical. Interestingly, Table 3.7 shows their ability to discern the difference becomes less acute when graphical animation is introduced.

3.4.2.5 Roles of Audio Realism and AV Synchronization

We did not include the results for the comparisons of the same clips in Tables 3.6 and 3.7, however, in each case close to 90% were able to detect the same video or audio clips. Earlier studies (van den Doel and Pai, 2002a; van den Doel, 2005) suggested that the subjects were not necessarily

able to detect the difference between single vs. multi-mode sounds or discern the same sounds when played again. Our simple test was designed to provide a calibration of our subject's ability to discern similar sounds in these sets of tests.

We can also see in Table 3.7 that subjects reliably preferred our method to those videos using manually synchronized, recorded sounds of varying quality. This study shows that simply adding sound effects to silent 3D animation of fluids does *not* automatically improve the perceived realism – the audio needs to be both realistic and seamlessly synchronized in order to improve the overall audio-visual experience.

3.4.2.6 Analysis

From this study, we see several interesting results. First, although we feel this work presents a significant step in computer synthesized sounds for liquids, the subjects still prefer real, recorded audio clips when no additional sound cues were generated, as shown in Table 3.4. Second, Table 3.5 shows that our method appears to consistently improve the perceived visual-audio experience – most significant in the case of interactive demos such as the rubber duck moving in a bath tub. Third, in side-by-side tests (Tables 3.6 and 3.7 top) for the audio only and audio-visual experiences, the subjects consistently prefer the sounds generated by our method over the sounds of single-sphere approximation. Finally, when audio is added to graphical animations (Table 3.7 bottom), the audio must be both realistic and synchronized seamlessly with the visual cues to improve the perceived realism of the overall experience.

3.5 Conclusion, Limitations, and Future Work

We present an automatic, physically-based synthesis method based on bubble resonance that generates liquid sounds directly from the fluid simulator. Our approach is general and applicable to different types of fluid simulation methods commonly used in computer graphics. It can run at interactive rates and its sound quality depends on the physical correctness of the fluid simulators. Our user study suggests that the perceived realism of liquid sounds generated using our approach is comparable to recorded sounds in similar settings.

Although our method generates adequately realistic sounds for multiple benchmarks, there are some limitations of our technique. Since we are generating sound from bubbles, the quality of the synthesized sounds depends on the accuracy and correctness of bubble formation from the fluid simulator. We also used a simplified model for the bubble excitation. Although no analytic solution exists, a more complex approximation could potentially help. Continued research on fluid simulations involving bubbles and bubble excitation would improve the quality and accuracy of the sound generated using our approach, specifically we expect that as fluid simulators are better able to generate the varied distribution of bubbles occuring in nature, the high frequency noise present in some of our demonstrations would be reduced.

For non-star-shaped bubbles, because they cannot be decomposed into spherical harmonics, we are forced to revert to the simple volume-based approximation. Since bubbles tend to be spherical (and rapidly become spherical without external forces), this happens rarely. It can, however, be see in the pouring water demo, when a ring-shaped bubble forms soon after the initial impact. There has been some recent work on simulating general bubble oscillations using a boundary element method (Pozrikidis, 2004) and we could provide more accuracy for complex bubble shapes using a similar technique, but not without substantially higher computational costs.

CHAPTER 4: EXAMPLE-GUIDED RIGID BODY SOUND SYNTHESIS

In this chapter, I discuss my work on example-guided rigid body sound synthesis. I begin with a dicussion of the mathematical background of modal sound syntehsis, the relationship between material properties and sounds, and the constraints of the material model that we used. After that, I describe the overall methodology of the simulation framework, followed by detailed discussions of individual stages: feature extraction, parameter estimation, and residual compensation. I then discribe the results obtained by my approach, as well as an analysis of the results. Finally, I conclude with a summary of my contributions and a discussion of possible future work.

4.1 Background

4.1.1 Modal Sound Synthesis:

The standard linear modal synthesis technique (Shabana, 1997) is frequently used for modeling of dynamic deformation and physically based sound synthesis. We adopt tetrahedral finite element models to represent any given geometry (O'Brien et al., 2002). The displacements, $\mathbf{x} \in \mathbb{R}^{3N}$, in such a system can be calculated with the following linear deformation equation:

$$\mathbf{M}\ddot{\mathbf{x}} + \mathbf{C}\dot{\mathbf{x}} + \mathbf{K}\mathbf{x} = \mathbf{f}, \tag{4.1}$$

where $\mathbf{M}$, $\mathbf{C}$, and $\mathbf{K}$ respectively represent the mass, damping and stiffness matrices. For small levels of damping, it is reasonable to approximate the damping matrix with *Rayleigh damping*, i.e. representing damping matrix as a linear combination of mass matrix and stiffness matrix: $\mathbf{C} = \alpha\mathbf{M} + \beta\mathbf{K}$. This is a well-established practice and has been adopted by many modal synthesis related works in both graphics and acoustics communities. After solving the generalized eigenvalue problem

$$\mathbf{K}\mathbf{U} = \mathbf{\Lambda}\mathbf{M}\mathbf{U}, \tag{4.2}$$

the system can be decoupled into the following form:

$$\ddot{\mathbf{q}} + (\alpha\mathbf{I} + \beta\mathbf{\Lambda})\dot{\mathbf{q}} + \mathbf{\Lambda}\mathbf{q} = \mathbf{U}^T\mathbf{f}, \tag{4.3}$$

where $\mathbf{\Lambda}$ is a diagonal matrix, containing the eigenvalues of Equation 4.2; $\mathbf{U}$ is the eigenvector matrix, and transforms $\mathbf{x}$ to the decoupled deformation bases $\mathbf{q}$ with $\mathbf{x} = \mathbf{U}\mathbf{q}$.

The solution to this decoupled system, Equation 4.3, are a bank of *modes*, i.e. damped sinusoidal waves. The i'th mode looks like:

$$q_i = a_i e^{-d_i t} \sin(2\pi f_i t + \theta_i), \tag{4.4}$$

where f_i is the frequency of the mode, d_i is the damping coefficient, a_i is the excited amplitude, and θ_i is the initial phase.

The frequency, damping, and amplitude together define the *feature* ϕ of mode i:

$$\phi_i = (f_i, d_i, a_i) \tag{4.5}$$

and will be used throughout the rest of the chapter. We ignore θ_i in Equation 4.4 because it can be safely assumed as zero in our estimation process, where the object is initially at rest and struck at $t = 0$. f and ω are used interchangeably to represent frequency, where $\omega = 2\pi f$.

4.1.2 Material properties

The values in Equation 4.4 depend on the material properties, the geometry, and the run-time interactions: a_i and θ_i depend on the run-time excitation of the object, while f_i and d_i depend on the geometry and the material properties as shown below. Solving Equation 4.3, we get

$$d_i = \frac{1}{2}(\alpha + \beta\lambda_i), \tag{4.6}$$

$$f_i = \frac{1}{2\pi}\sqrt{\lambda_i - \left(\frac{\alpha + \beta\lambda_i}{2}\right)^2}. \tag{4.7}$$

We assume the Rayleigh damping coefficients, α and β, can be transfered to another object with no drastic shape or size change. Empirical experiments were carried out to support this assumption.

Please refer to (Ren et al., 2012) for more detail. The eigenvalues λ_i's are calculated from **M** and **K** and determined by the geometry and tetrahedralization as well as the material properties: in our tetrahedral finite element model, **M** and **K** depend on mass density ρ, Young's modulus E, and Poisson's ratio ν, if we assume the material is *isotropic* and *homogeneous*.

4.1.3 Constraint for modes

We observe modes in the adopted linear modal synthesis model have to obey some constraint due to its formulation. Because of the Rayleigh damping model we adopted, all estimated modes lie on a circle in the (ω, d)-space, characterized by α and β. This can be shown as follows. Rearranging Equation 4.6 and Equation 4.7 as

$$\omega_i{}^2 + \left(d_i - \frac{1}{\beta}\right)^2 = \left(\frac{1}{\beta}\sqrt{1 - \alpha\beta}\right)^2 \tag{4.8}$$

we see that it takes the form of $\omega_i{}^2 + (d_i - y_c)^2 = R^2$. This describes a circle of radius R centered at $(0, y_c)$ in the (ω, d)-space, where R and y_c depend on α and β. This constraint for modes restricts the model from capturing some sound effects and renders it impossible to make modal synthesis sounds with Rayleigh damping exactly the same as an arbitrary real-world recording. However, if a circle that best represents the recording audio is found, it is possible to preserve the same sense of material as the recording. It is shown in Section 4.3 and 4.4.3, how a proposed pipeline achieves this.

4.2 Methodology

Figure 4.1 shows an example of our framework. From one recorded impact sound (Figure 4.1a), we estimated material parameters, which can be directly applied to various geometries (Figure 4.1c, 4.1d, 4.1e) to generate audio effects that automatically reflect the shape variation while still preserve the same sense of material. Figure 4.2 depicts the pipeline of our approach, and its various stages are explained below.

Feature extraction: Given a recorded impact audio clip, from which we first extract some high-level *features*, namely, a set of damped sinusoids with constant frequencies, dampings, and initial amplitudes (Sec. 4.3). These features are then used to facilitate estimation of the material parameters (Sec. 4.4), and guide the residual compensation process (Sec. 4.5).

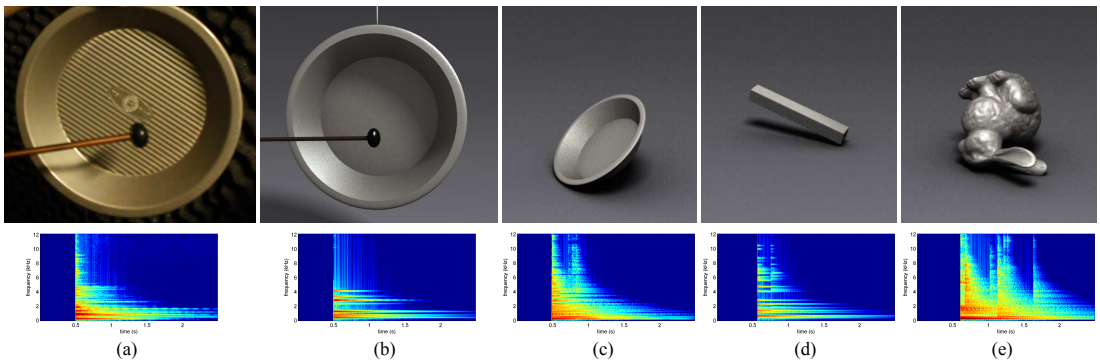


Figure 4.1: From the recording of a real-world object (a), our framework is able to find the material parameters and generates similar sound for a replicate object (b). The same set of parameters can be transfered to various virtual objects to produce sounds with the same material quality ((c), (d), (e)).

Parameter estimation: Due to the constraints of the sound synthesis model, we assume a limited input from just one recording and it is challenging to estimate the material parameters from one audio sample. To do so, a virtual object of the same size and shape as the real-world object used in recording the example audio is created. Each time an estimated set of parameters are applied to the virtual object for a given impact, the generated sound, as well as the feature information of the resonance modes, are compared with the real world example sound and extracted features respectively using a difference metric. This metric is designed based on *psychoacoustic* principles, and aimed at measuring both the audio material resemblance of two objects and the perceptual similarity between two sound clips. The optimal set of material parameters is thereby determined by minimizing this perceptually inspired metric function (see Sec. 4.4). These parameters are readily transferable to other virtual objects of various geometries undergoing rich interactions, and the synthesized sounds preserve the intrinsic quality of the original sounding material.

Residual compensation: Finally, our approach also accounts for the residual, i.e. the approximated differences between the real-world audio recording and the modal synthesis sound with the estimated parameters. First, the residual is computed using the extracted features, the example recording, and the synthesized audio. Then at run-time, the residual is transfered to various virtual objects. The transfer of residual is guided by the transfer of modes, and naturally reflects the geometry and run-time interaction variation (see Sec. 4.5).

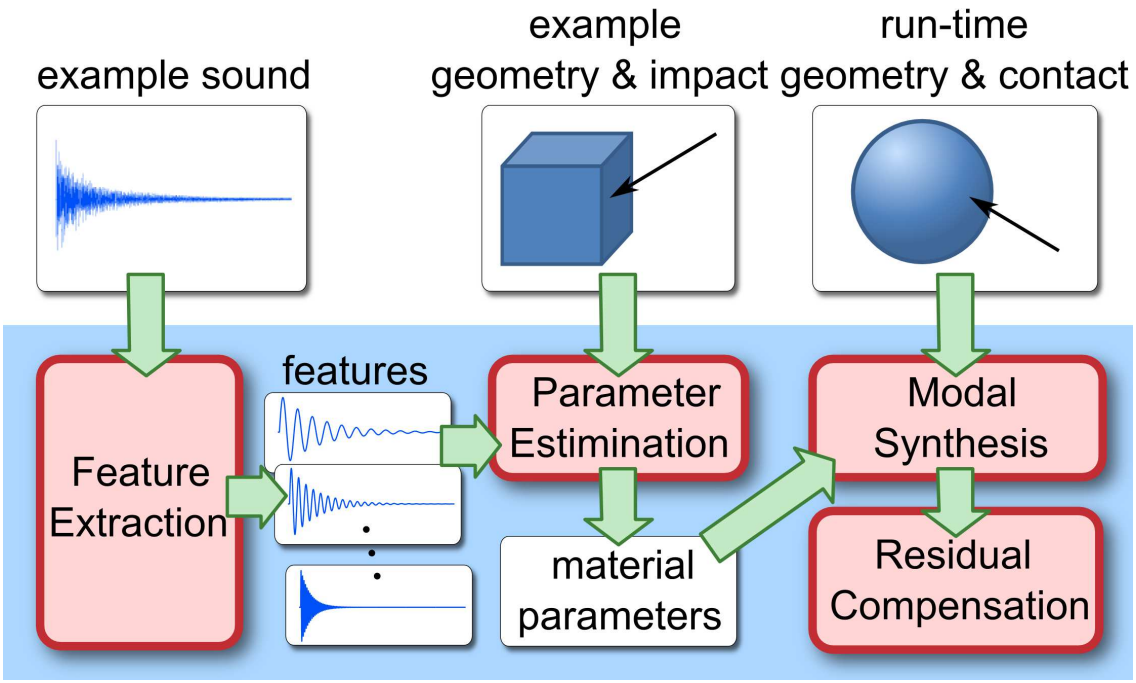


Figure 4.2: Overview of the example-guided sound synthesis framework (shown in the blue block): Given an example audio clip as input, features are extracted. They are then used to search for the optimal material parameters based on a perceptually inspired metric. A residual between the recorded audio and the modal synthesis sound is calculated. At run-time, the excitation is observed for the modes. Corresponding rigid-body sounds that have a similar audio quality as the original sounding materials can be automatically synthesized. A modified residual is added to generate a more realistic final sound.

4.3 Feature Extraction

An example impact sound can be represented by high-level features collectively.

We first analyze and decompose a given example audio clip into a set of features, which will later be used in the subsequent phases of our pipeline, namely the parameter estimation and residual compensation parts. Next we present the detail of our feature extraction algorithm.

Multi-level power spectrogram representation: As shown in Equation 4.5, the feature of a mode is defined as its frequency, damping, and amplitude. In order to analyze the example audio and extract these feature values, we use a time-varying frequency representation called *power spectrogram*. A power spectrogram $\mathbf{P}$ for a a time domain signal $\mathbf{s}[n]$, is obtained by first breaking it up into

overlapping frames, and then performing windowing and Fourier transform on each frame:

$$\mathbf{P}[m,\omega] = \left| \sum_{n} \mathbf{s}[n]\mathbf{w}[n-m]e^{-j\omega n} \right|^2, \tag{4.9}$$

where $\mathbf{w}$ is the window applied to the original time domain signal (Oppenheim et al., 1989). The power spectrogram records the signal's power spectral density within a *frequency bin* centered around $\omega = 2\pi f$ and a *time frame* defined by m.

When computing the power spectrogram for a given sound clip, one can choose the resolutions of the time or frequency axes by adjusting the length of the window $\mathbf{w}$. Choosing the resolution in one dimension, however, automatically determines the resolution in the other dimension. A high frequency resolution results in a low temporal resolution, and vice versa.

To fully accommodate the range of frequency and damping for all the modes of an example audio, we compute multiple levels of power spectrograms, with each level doubling the frequency resolution of the previous one and halving the temporal resolution. Therefore, for each mode to be extracted, a suitable level of power spectrogram can be chosen first, depending on the time and frequency characteristics of the mode.

Global-to-local scheme: After computing a set of multi-level power spectrograms for a recorded example audio, we *globally* search through all levels for peaks (local maxima) along the frequency axis. These peaks indicate the frequencies where potential modes are located, some of which may appear in multiple levels. At this step the knowledge of frequency is limited by the frequency resolution of the level of power spectrogram. For example, in the level where the window size is 512 points, the frequency resolution is as coarse as 86 Hz. A more accurate estimate of the frequency as well as the damping value is obtained by performing a *local shape fitting* around the peak.

The power spectrogram of a damped sinusoid has a 'hill' shape, similar to the blue surface shown in Figure 4.3b. The actual shape contains information of the damped sinusoid: the position and height of the peak are respectively determined by the frequency and amplitude, while the slope along the time axis and the width along the frequency axis are determined by the damping value. For a potential mode, a damped sinusoid with the initial guess of (f, d, a) is synthesized and added to the sound clip consisting of all the modes collected so far. The power spectrogram of the resulting sound clip is computed (shown as the red hill shape in Figure 4.3b), and compared locally with that of the

recorded audio (the blue hill shape in Figure 4.3b)). An optimizer then searches in the continuous (f, d, a)-space to minimize the difference and acquire a refined estimate of the frequency, damping, and amplitude of the mode at question. Figure 4.3 illustrates this process.

The local shape fittings for all potential modes are performed in a greedy manner. Among all peaks in all levels, the algorithm starts with the one having the highest average power spectral density. If the shape fitting error computed is above a predefined threshold, we conclude that this level of power spectrogram is not sufficient in capturing the feature characteristics and thereby discard the result; otherwise the feature of the mode is collected. In other words, the most suitable time-frequency resolution (level) for a mode with a particular frequency is not predetermined, but dynamically searched for. Similar approaches have been proposed to analyze the sinusoids in an audio clip in a multi-resolution manner (e.g. Levine et al. (1998), where the time-frequency regions' power spectrogram resolution is predetermined).

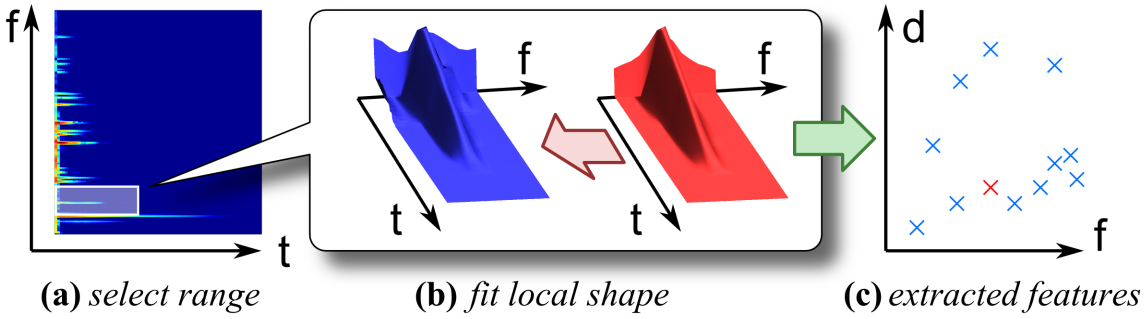


Figure 4.3: Feature extraction from a power spectrogram. (a) A peak is detected in a power spectrogram at the location of a potential mode. f=frequency, t=time. (b) A local shape fitting of the power spectrogram is performed to estimate the frequency, damping and amplitude of the potential mode. (c) If the fitting error is below a certain threshold, we collect it in the set of extracted features, shown as the red cross in the feature space. (Only the frequency f and damping d are shown here.)

We have tested the accuracy of our feature extraction with 100 synthetic sinusoids with frequencies and damping values randomly drawn from $[0, 22050.0]$(Hz) and $[0.1, 1000](s^{-1})$ respectively. The average relative error is 0.040% for frequencies and 0.53% for damping values, which are sufficient for our framework.

Comparison with existing methods: The SMS method (Serra and Smith III, 1990) is also capable of estimating information of modes. From a power spectrogram, it tracks the amplitude envelope of each peak over time, and a similar method is adopted by Lloyd et al. (2011). Unlike our algorithm,

which fits the entire local hill shape, they only track a single peak value per time frame. In the case where the mode's damping is high or the signal's background is noisy, this method yields high error.

Another feature extraction technique was proposed by Pai et al. (2001) and Corbett et al. (2007). The method is known for its ability to separate modes within one frequency bin. In our framework, however, the features are only used to guide the subsequent parameter estimation process, which is not affected much by replacing two nearly duplicate features with one. Our method also offers some advantages and achieves higher accuracy in some cases compared with theirs. First, our proposed greedy approach is able to reduce the interference caused by high energy neighboring modes. Secondly, these earlier methods use a fixed frequency-time resolution that is not necessarily the most suitable for extracting all modes, while our method selects the appropriate resolution dynamically.

The detailed comparisons and data can be found in Sec 4.6.1.

4.4 Parameter Estimation

Using the extracted features (Sec. 4.3) and psychoacoustic principles (as described in this section), we introduce a parameter estimation algorithm based on an optimization framework for sound synthesis.

4.4.1 An Optimization Framework

We now describe the optimization work flow for estimating material parameters for sound synthesis. In the rest of the chapter, all data related to the example audio recordings are called *reference* data; all data related to the virtual object (which are used to estimate the material parameters) are called *estimated* data, and are denoted with a tilde, e.g. $\tilde{f}$.

Reference sound and features: The *reference sound* is the example recorded audio, which can be expressed as a time domain signal $\mathbf{s}[n]$. The *reference features* $\mathbf{\Phi} = \{\phi_i\} = \{(f_i, d_i, a_i)\}$ are the features extracted from the reference sound, as described in Sec. 4.3.

Estimated sound and features: In order to compute the *estimated sound* $\tilde{\mathbf{s}}[n]$ and *estimated features* $\tilde{\mathbf{\Phi}} = \{\tilde{\phi}_j\} = \{(\tilde{f}_j, \tilde{d}_j, \tilde{a}_j)\}$, we first create a virtual object that is roughly the same size and geometry as the real-world object whose impact sound was recorded. We then tetrahedralize it and calculate its

mass matrix $\mathbf{M}$ and stiffness matrix $\mathbf{K}$. As mentioned in Sec. 4.1, we assume the material is isotropic and homogeneous. Therefore, the initial $\mathbf{M}$ and $\mathbf{K}$ can be found using the finite element method, by assuming some initial values for the Young's modulus, mass density, and Poisson's ratio, E_0, ρ_0, and ν_0. The assumed eigenvalues λ_i^0's can thereby be computed. For computational efficiency, we make a further simplification that the Poisson's ratio is held as constant. Then the eigenvalue λ_i for general E and ρ is just a multiple of λ_i^0:

$$\lambda_i = \frac{\gamma}{\gamma_0}\lambda_i^0 \tag{4.10}$$

where $\gamma = E/\rho$ is the ratio of Young's modulus to density, and $\gamma_0 = E_0/\rho_0$ is the ratio using the assumed values.

We then apply a unit impulse on the virtual object at a point corresponding to the actual impact point in the example recording, which gives an excitation pattern of the eigenvalues as Equation 4.4. We denote the excitation amplitude of mode j as a_j^0. The superscript 0 notes that it is the response of a unit impulse; if the impulse is not unit, then the excitation amplitude is just scaled by a factor σ,

$$a_j = \sigma a_j^0 \tag{4.11}$$

Combining Equation 4.6, Equation 4.7, Equation4.10, and Equation4.11, we obtain a mapping from an assumed eigenvalue and its excitation (λ_j^0, a_j^0) to an estimated mode with frequency $\tilde{f}_j$, damping $\tilde{d}_j$, and amplitude $\tilde{a}_j$:

$$(\lambda_j^0, a_j^0) \xrightarrow{\{\alpha,\beta,\gamma,\sigma\}} (\tilde{f}_j, \tilde{d}_j, \tilde{a}_j). \tag{4.12}$$

The estimated sound $\tilde{\mathbf{s}}[n]$, is thereby generated by mixing all the estimated modes,

$$\tilde{\mathbf{s}}[n] = \sum_j \left(\tilde{a}_j e^{-\tilde{d}_j(n/F_s)} \sin(2\pi\tilde{f}_j(n/F_s))\right) \tag{4.13}$$

where F_s is the sampling rate.

Difference metric: The estimated sound $\tilde{\mathbf{s}}[n]$ and features $\tilde{\mathbf{\Phi}}$ can then be compared against the reference sound $\mathbf{s}[n]$ and features $\mathbf{\Phi}$, and a difference metric can be computed. If such difference

metric function is denoted by Π, the problem of parameter estimation becomes finding

$$\{\alpha, \beta, \gamma, \sigma\} = \underset{\{\alpha,\beta,\gamma,\sigma\}}{\arg\min}\, \Pi. \tag{4.14}$$

An optimization process is used to find such parameter set. The most challenging part of our work is to find a suitable metric function that can truly reflect what we view as the difference. Next we discuss the details about the metric design in Sec. 4.4.2 and the optimization process in Sec. 4.4.3.

4.4.2 Metric

Given an impact sound of a real-world object, the goal is to find a set of material parameters such that when they are applied to a virtual object of the same size and shape, the synthesized sounds have the similar auditory perception as the original recorded sounding object. By further varying the size, geometry, and the impact points of the virtual object, the intrinsic 'audio signature' of each material for the synthesized sound clips should closely resemble that of the original recording. These are the two criteria guiding the estimation of material parameters based on an example audio clip:

1. the perceptual similarity of two sound clips;

2. the audio material resemblance of two generic objects.

The perceptual similarity of sound clips can be evaluated by an 'image domain metric' quantified using the power spectrogram; while the audio material resemblance is best measured by a 'feature domain metric' – both will be defined below,

Image domain metric: Given a reference sound $\mathbf{s}[n]$ and an estimated sound $\tilde{\mathbf{s}}[n]$, their power spectrograms are computed using Equation 4.9 and denoted as two 2D images: $\mathbf{I} = \mathbf{P}[m, \omega]$, $\tilde{\mathbf{I}} = \tilde{\mathbf{P}}[m, \omega]$. An image domain metric can then be expressed as

$$\Pi_{image}(\mathbf{I}, \tilde{\mathbf{I}}). \tag{4.15}$$

Our goal is to find an estimated image $\tilde{\mathbf{I}}$ that minimizes a given image domain metric. This process is equivalent to image registration in computer vision and medical imaging.

Feature domain metric: A feature $\phi_i = (f_i, d_i, a_i)$ is essentially a three dimensional point. As established in Sec. 4.1, the set of features of a sounding object is closely related to the material

properties of that object. Therefore a metric defined in the feature space is useful in measuring the audio material resemblance of two objects. In other words, a good estimate of material parameters should map the eigenvalues of the virtual object to similar modes as that of the real object. A feature domain metric can be written as

$$\Pi_{feature}(\mathbf{\Phi}, \tilde{\mathbf{\Phi}}) \tag{4.16}$$

and the process of finding the minimum can be viewed as a point set matching problem in computer vision.

Hybrid metric: Both the auditory perceptual similarity and audio material resemblance would need to be considered for a generalized framework, in order to extract and transfer material parameters for modal sound synthesis using a recorded example to guide the automatic selection of material parameters. Therefore, we propose a novel 'hybrid' metric that takes into account of both:

$$\Pi_{hybrid}(\mathbf{I}, \mathbf{\Phi}, \tilde{\mathbf{I}}, \tilde{\mathbf{\Phi}}). \tag{4.17}$$

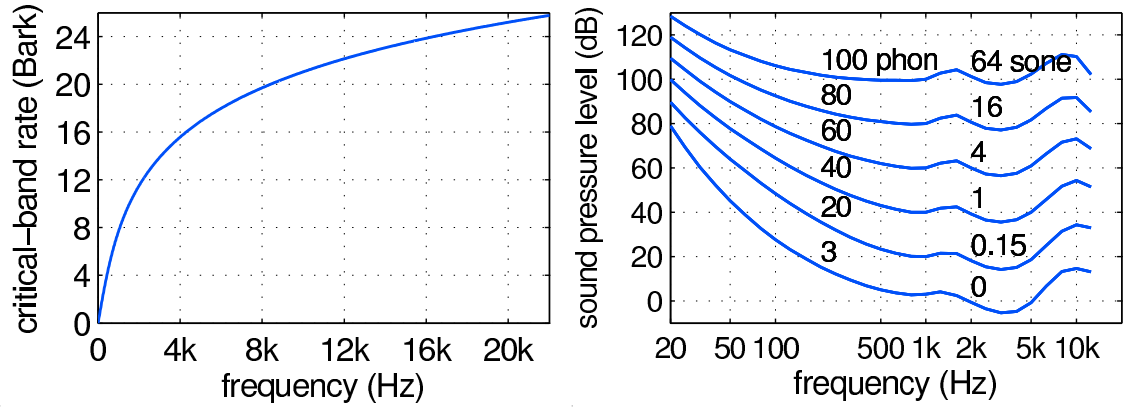
Next, we provide details on how we design and compute these metrics.

4.4.2.1 Image Domain Metric

Given two power spectrogram images $\mathbf{I}$ and $\tilde{\mathbf{I}}$, a naive metric can be defined as their squared difference: $\Pi_{image}(\mathbf{I}, \tilde{\mathbf{I}}) = \sum_{m,\omega} \left(\mathbf{P}[m, \omega] - \tilde{\mathbf{P}}[m, \omega]\right)^2$. There are, however, several problems with this metric. The frequency resolution is uniform across the spectrum, and the intensity is uniformly weighted. As humans, however, we distinguish lower frequencies better than the higher frequencies, and mid-frequency signals appear louder than extremely low or high frequencies (Zwicker and Fastl, 1999). Therefore, directly taking squared difference of power spectrograms overemphasizes the frequency differences in the high-frequency components and the intensity differences near both ends of the audible frequency range. It is necessary to apply both *frequency* and *intensity* transformations before computing the image domain metric. We design these transformations based on psychoacoustic principles (Zwicker and Fastl, 1999).

Frequency transformation: Studies in psychoacoustics suggested that humans have a limited capacity to discriminate between nearby frequencies, i.e. a frequency f_1 is not distinguishable from

f_2 if f_2 is within $f_1 \pm \Delta f$. The indistinguishable range Δf is itself a function of frequency, for example, the higher the frequency, the larger the indistinguishable range. To factor out this variation in Δf a different frequency representation, called *critical-band rate* z, has been introduced in psychoacoustics. The unit for z is *Bark*, and it has the advantage that while Δf is a function of f (measured in Hz), it is constant when measured in Barks. Therefore, by transforming the frequency dimension of a power spectrogram from f to z, we obtain an image that is weighted according to human's perceptual frequency differences. Figure 4.4a shows the relationship between critical-band rate z and frequency f, $z = Z(f)$.



(a) (b)

Figure 4.4: **Psychoacoustics related values:** (a) the relationship between critical-band rate (in Bark) and frequency (in Hz); (b) the relationship between loudness level L_N (in phon), loudness L (in sone), and sound pressure level L_p (in dB). Each curve is an *equal-loudness contour*, where a constant loudness is perceived for pure steady tones with various frequencies.

Intensity transformation: Sound can be described as the variation of pressure, $p(t)$, and human auditory system has a high dynamical range, from 10^{-5} Pa (threshold of hearing) to 10^2 Pa (threshold of pain). In order to cope with such a broad range, the *sound pressure level* is normally used. For a sound with pressure p, its sound pressure level L_p in decibel (abbreviated to dB-SPL) is defined as

$$L_p = 20 \log(p/p_0), \tag{4.18}$$

where p_0 is a standard reference pressure. While L_p is just a physical value, *loudness* L is a perceptual value, which measures human sensation of sound intensity. In between, *loudness level* L_N relates the physical value to human sensation. Loudness level of a sound is defined as the sound pressure level

of a 1-kHz tone that is perceived as loud as the sound. Its unit is *phon*, and is calibrated such that a sound with loudness level of 40 phon is as loud as a 1-kHz tone at 40 dB-SPL. Finally, loudness L is computed from loudness level. Its unit is *sone*, and is defined such that a sound of 40 phon is 1 sone; a sound twice as loud is 2 sone, and so on.

Figure 4.4b shows the relationship between sound pressure level L_p, loudness level L_N and loudness L according to the international standard (ISO, 2003). The curves are *equal-loudness contours*, which are defined such that for different frequency f and sound pressure level L_p, the perceived loudness level L_N and loudness L is constant along each equal-loudness contour. Therefore the loudness of a signal with a specific frequency f and sound pressure level L_p can be calculated by finding the equal-loudness contour passing (f, L_p).

There are other psychoacoustic factors that can affect the human sensation of sound intensity. For example, van den Doel et al. (van den Doel and Pai, 2002b; van den Doel et al., 2004) considered the 'masking' effect, which describes the change of audible threshold in the presence of multiple stimuli, or modes in this case. However, they did not handle the loudness transform above the audible threshold, which is critical in our perceptual metric. Similar to the work by van den Doel and Pai (1998), we have ignored the masking effect.

Psychoacoustic metric: After transforming the frequency f (or equivalently, ω) to the critical-band rate z and mapping the intensity to loudness, we obtain a transformed image $\mathbf{T}(\mathbf{I}) = \mathbf{T}(\mathbf{I})[m, z]$. Different representations of a sound signal is shown in Figure 4.5. Then we can define a psychoacoustic image domain metric as

$$\Pi_{psycho}(\mathbf{I}, \tilde{\mathbf{I}}) = \sum_{m,z} \left(\mathbf{T}(\mathbf{I})[m, z] - \mathbf{T}(\tilde{\mathbf{I}})[m, z]\right)^2 \tag{4.19}$$

Similar transformations and distance measures have also been used to estimate the perceived resemblance between music pieces (Morchen et al., 2006; Pampalk et al., 2002).

4.4.2.2 Feature Domain Metric

As shown in Equation 4.8, in the (ω, d)-space, modes under the assumption of Rayleigh damping lie on a circle determined by damping parameters α and β, while features extracted from example recordings can be anywhere. Therefore, it is challenging to find a good match between the reference

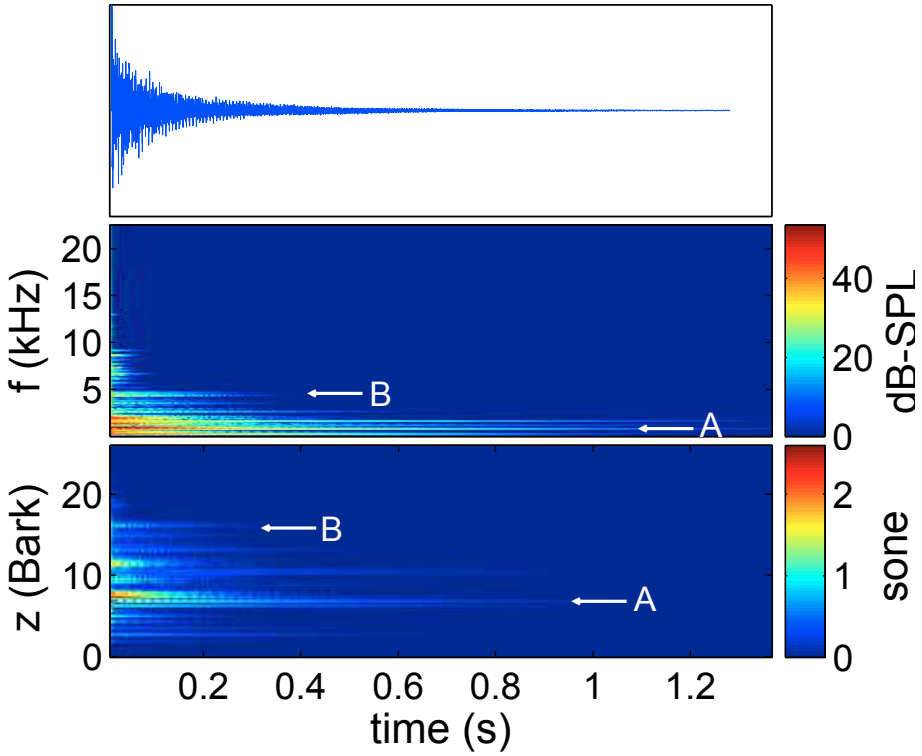


Figure 4.5: Different representation of a sound clip. Top: time domain signal $\mathbf{s}[\mathbf{n}]$. Middle: original image, power spectrogram $P[m, \omega]$ with intensity measured in dB. Bottom: image transformed based on psychoacoustic principles. The frequency f is transformed to *critical-band rate* z, and the intensity is transformed to *loudness*. Two pairs of corresponding modes are marked as A and B. It can be seen that the frequency resolution decreases toward the high frequencies, while the signal intensities in both the higher- and lower-end of the spectrum are de-emphasized.

features Φ and estimated features $\tilde{\Phi}$. Figure 4.6a shows a typical matching in the (f, d)-space. Next we present a feature domain metric that evaluates such a match.

In order to compute the feature domain metric, we first transform the frequency and damping of feature points to another different 2D space. Namely, from (f_i, d_i) to (x_i, y_i), where $x_i = \mathrm{X}(f_i)$ and $y_i = \mathrm{Y}(d_i)$ encode the frequency and damping information respectively. With suitable transformations, the Euclidean distance defined in the transformed space can be more useful and meaningful for representing the perceptual difference. The distance between two feature points is thus written as

$$D(\phi_i, \tilde{\phi}_j) \equiv \left\| \Big(\mathrm{X}(f_i), \mathrm{Y}(d_i)\Big) - \Big(\mathrm{X}(\tilde{f}_j), \mathrm{Y}(\tilde{d}_j)\Big) \right\|. \tag{4.20}$$

Frequency and damping are key factors in determining material agreement, while amplitude indicates relative importance of modes. That is why we measure the distance between two feature points in the 2D (f, d)-space and use amplitude to weigh that distance.

For frequency, as described in Sec. 4.4.2.1 we know that the frequency resolution of human is constant when expressed as critical-band rate and measured in Barks: $\Delta f(f) \propto \Delta z$. Therefore it is a suitable frequency transformation

$$X(f) = c_z Z(f) \tag{4.21}$$

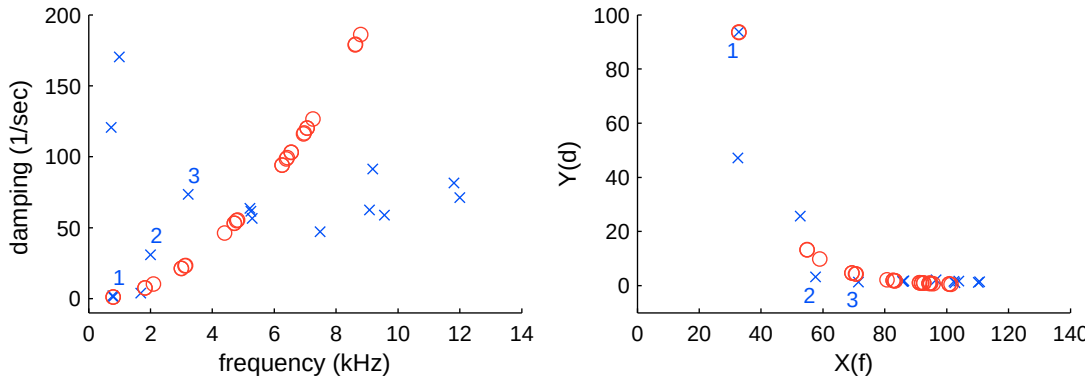
where c_z is some constant coefficient.

For damping, although human can roughly sense that one mode damps faster than another, directly taking the difference in damping value d is not feasible. This is due to the fact that humans cannot distinguish between extremely short bursts (Zwicker and Fastl, 1999). For a damped sinusoid, the inverse of the damping value, $1/d_i$, is proportional to its duration, and equals to how long before the signal decays to e^{-1} of its initial amplitude. While distance measured in damping values overemphasizes the difference between signals with high d values (corresponding to short bursts), distance measured in durations does not. Therefore

$$Y(d) = c_d \frac{1}{d} \tag{4.22}$$

(where c_d is some constant coefficient) is a good choice of damping transformation. The reference and estimated features of data in Figure 4.6a are shown in the transformed space in Figure 4.6b.

Having defined the transformed space, we then look for matching the reference and estimated feature points in this space. Our matching problem belongs to the category where there is no known correspondence, i.e. no prior knowledge about which point in one set should be matched to which point in another. Furthermore, because there may be several estimated feature points in the neighborhood of a reference point or vice versa, the matching is not necessarily a one-to-one relationship. There is also no guarantee that an exact matching exist, because (1) the recorded material may not obey the Rayleigh damping model, (2) the discretization of the virtual object and the assumed hit point may not give the exact eigenvalues and excitation pattern of the real object. Therefore we are merely looking for a partial, approximate matching.



(a) (b)

Figure 4.6: Point set matching problem in the feature domain: (a) in the original frequency and damping, (f, d)-space. (b) in the transformed, (x, y)-space, where $x = X(f)$ and $y = Y(d)$. The blue crosses and red circles are the reference and estimated feature points respectively. The three features having the largest energies are labeled 1, 2, and 3.

The simplest point-based matching algorithm that solves problems in this category (i.e. partial, approximate matching without known correspondence) is Iterative Closest Points. It does not work well, however, when there is a significant number of feature points that cannot be matched (Besl and McKay, 1992), which is possibly the case in our problem. Therefore, we define a metric, *Match Ratio Product*, that meets our need and is discussed next.

For a reference feature point set $\mathbf{\Phi}$, we define a *match ratio* that measures how well they are matched by an estimated feature point set $\tilde{\mathbf{\Phi}}$. This *set-to-set* match ratio, defined as

$$R(\mathbf{\Phi}, \tilde{\mathbf{\Phi}}) = \frac{\sum_i w_i R(\phi_i, \tilde{\mathbf{\Phi}})}{\sum_i w_i}, \tag{4.23}$$

is a weighted average of the *point-to-set* match ratios, which are in turn defined as

$$R(\phi_i, \tilde{\mathbf{\Phi}}) = \frac{\sum_j \tilde{u}_{ij} k(\phi_i, \tilde{\phi}_j)}{\sum_j \tilde{u}_{ij}}, \tag{4.24}$$

a weighted average of the *point-to-point* match scores $k(\phi_i, \tilde{\phi}_j)$. The point-to-point match score $k(\phi_i, \tilde{\phi}_j)$, which is directly related to the distance of feature points (Equation 4.20), should be designed to give values in the continuous range $[0, 1]$, with 1 meaning that the two points coincide, and 0 meaning that they are too far apart. Similarly $R(\phi_i, \tilde{\mathbf{\Phi}}) = 1$ when ϕ_i coincides with an estimated feature point, and $R(\mathbf{\Phi}, \tilde{\mathbf{\Phi}}) = 1$ when all reference feature points are perfectly matched. The weight

w_i and $\tilde{u}_{ij}$ in Equation 4.23 and Equation 4.24 are used to adjust the influence of each mode. The match ratio for the estimated feature points, $\tilde{R}$, is defined analogously

$$\tilde{R}(\mathbf{\Phi}, \tilde{\mathbf{\Phi}}) = \frac{\sum_j \tilde{w}_j R(\tilde{\phi}_j, \mathbf{\Phi})}{\sum_i \tilde{w}_j} \tag{4.25}$$

The match ratios for the reference and the estimated feature point sets are then combined to form the *Match Ratio Product* (MRP), which measures how well the reference and estimated feature point sets match with each other,

$$\Pi_{MRP}(\mathbf{\Phi}, \tilde{\mathbf{\Phi}}) = -R\tilde{R}. \tag{4.26}$$

The negative sign is to comply with the minimization framework. Multiplying the two ratios penalizes the extreme case where either one of them is close to zero (indicating poor matching).

The normalization processes in Equation 4.23 and Equation 4.25 are necessary. Notice that the denominator in Equation 4.25 is related to the number of estimated feature points inside the audible range, $\tilde{N}_{\text{audible}}$ (in fact $\sum_j \tilde{w}_j = \tilde{N}_{\text{audible}}$ if all $\tilde{w}_j = 1$). Depending on the set of parameters, $\tilde{N}_{\text{audible}}$ can vary from a few to thousands. Factoring out $\tilde{N}_{\text{audible}}$ prevents the optimizer from blindly introducing more modes into the audible range, which may increase the absolute number of matched feature points, but may not necessarily increase the match ratios. Such averaging techniques have also been employed to improve the robustness and discrimination power of point-based object matching methods (Dubuisson and Jain, 1994; Gope and Kehtarnavaz, 2007).

In practice, the weights w's and u's, can be assigned according to the relative energy or perceptual importance of the modes. The point-to-point match score $k(\phi_i, \tilde{\phi}_j)$, can also be tailored to meet different needs. The constants and function forms used in this section are listed in Sec 4.5.2.3.

4.4.2.3 Hybrid Metric

Finally, we combine the strengths from both image and feature domain metrics by defining the following hybrid metric:

$$\Pi_{hybrid} = \frac{\Pi_{psycho}}{|\Pi_{MRP}|}. \tag{4.27}$$

This metric essentially weights the perceptual similarity with how well the features match, and by making the match ratio product as the denominator, we ensure that a bad match (low MRP) will boost the metric value and is therefore highly undesirable.

4.4.3 Optimizer

We use the Nelder-Mead method (Lagarias et al., 1999) to minimize Equation 4.14, which may converge into one of the many local minima. We address this issue by starting the optimizer from many starting points, generated based on the following observations.

First, as elaborated by Equation 4.8 in Sec. 4.1, the estimated modes are constrained by a circle in the (ω, d)-space. Secondly, although there are many reference modes, they are not evenly excited by a given impact– we observe that usually the energy is mostly concentrated in a few dominant ones. Therefore, a good estimate of α and β must define a circle that passes through the neighborhood of these dominant reference feature points. We also observe that in order to yield a low metric value, there must be at least one dominant estimated mode at the frequency of the *most* dominant reference mode.

We thereby generate our starting points by first drawing two dominant reference feature points from a total of $N_{dominant}$ of them, and find the circle passing through these two points. This circle is potentially a 'good' circle, from which we can deduce a starting estimate of α and β using Equation 4.8. We then collect a set of eigenvalues and amplitudes (defined in Sec. 4.4.1) $\{(\lambda_j^0, a_j^0)\}$, such that there does not exist any (λ_k^0, a_k^0) that simultaneously satisfies $\lambda_k^0 < \lambda_j^0$ and $a_k^0 > a_j^0$. It can be verified that the estimated modes mapped from this set always includes the one with the highest energy, for any mapping parameters $\{\alpha, \beta, \gamma, \sigma\}$ used in Equation 4.12. Each (λ_j^0, a_j^0) in this set is then mapped and aligned to the frequency of the most dominant reference feature point, and its amplitude is adjusted to be identical as the latter. This step gives a starting estimate of γ and σ. Each set of $\{\alpha, \beta, \gamma, \sigma\}$ computed in this manner is a starting point, and may lead to a different local minimum. We choose the set which results in the lowest metric value to be our estimated parameters. Although there is no guarantee that a global minimum will be met, we find that the results produced with this strategy are satisfactory in our experiments, as discussed in Sec. 4.6.

4.5 Residual Compensation

With the optimization proposed in Sec. 4.4, a set of parameters that describe the material of a given sounding object can be estimated, and the produced sound bears a close resemblance of the material used in the given example audio. However, linear modal synthesis alone is not capable of synthesizing sounds that are as rich and realistic as many real-world recordings. Firstly, during the short period of contact, not all energy is transformed into stable vibration that can be represented with a small number of damped sinusoids, or modes. The stochastic and transient nature of the non-modal components makes sounds in nature rich and varying. Secondly, as discussed in Sec. 4.1, not all features can be captured due to the constraints for modes in the synthesis model. In this section we present a method to account for the *residual*, which approximates the difference between the real-world recordings and the modal synthesis sounds. In addition, we propose a technique for transferring the residual with geometry and interaction variation. With the residual computation and transfer algorithms introduced below, more realistic sounds that automatically vary with geometries and hitting points can be generated with a small computation overhead.

4.5.1 Residual Computation

In this section we discuss how to compute the residual from the recorded sound and the synthesized modal sound generated with the estimated parameters.

Previous works have also looked into capturing the difference between a source audio and its modal component (Serra and Smith III, 1990; Serra, 1997; Lloyd et al., 2011). In these works, the modal part is directly tracked from the original audio, so the residual can be calculated by a straightforward subtraction of the power spectrograms. The synthesized modal sound in our framework, however, is generated solely from the estimated material parameters. Although it preserves the intrinsic quality of the recorded material, in general the modes in our synthesized sounds are not perfectly aligned with the recorded audio. An example is shown in Figure 4.7a and Figure 4.7c. It is due to the constraints in our sound synthesis model and discrepancy between the discretized virtual geometries and the real-world sounding objects. As a result, direct subtraction does not work in this case to generate a reasonable residual. Instead, we first compute an intermediate data, called the *represented sound*. It corresponds to the part in the recorded sound that is captured,

or represented, by our synthesized sound. This represented sound (Figure 4.7d) can be directly subtracted from the recorded sound to compute the residual (Figure 4.7e).

The computation of the represented sound is based on the following observations. Consider a feature (described by ϕ_i) extracted from the recorded audio. If it is perfectly captured by the estimated modes, then it should not be included in the residual and should be completely subtracted from the recorded sound. If it is not captured at all, it should not be subtracted from the recorded sound, and if it is approximated by an estimated mode, it should be partially subtracted. Since features closely represent the original audio, they can be directly subtracted from the recorded sound.

The point-to-set match ratio $R(\phi_i, \tilde{\mathbf{\Phi}})$ proposed in Sec. 4.4.2 essentially measures how well a reference feature ϕ_i is represented (matched) by all the estimated modes. This match ratio can be conveniently used to determine how much of the corresponding feature should be subtracted from the recording.

The represented sound is therefore obtained by adding up all the reference features that are respectively weighted by the match ratio of the estimated modes. And the power spectrogram of the residual is obtained by subtracting the power spectrogram of the represented sound from that of the recorded sound. Figure 4.7 illustrates the residual computation process.

4.5.2 Residual Transfer

Residual of one particular instance (i.e. one geometry and one hit point) can be obtained through the above described residual computation method. However, when synthesizing sounds for a different geometry undergoing different interaction with other rigid bodies, the residual audio needs to vary accordingly. Lloyd et al. (2011) proposed applying a random dip filter on the residual to provide variation. While this offers an attractive solution for quickly generating modified residual sound, it does not transfer accordingly with the geometry change or the dynamics of the sounding object.

4.5.2.1 Algorithm

As discussed in previous sections, *modes* transfer naturally with geometries in the modal analysis process, and they respond to excitations at runtime in a physical manner. In other words, the modal component of the synthesized sounds already provides transferability of sounds due to varying

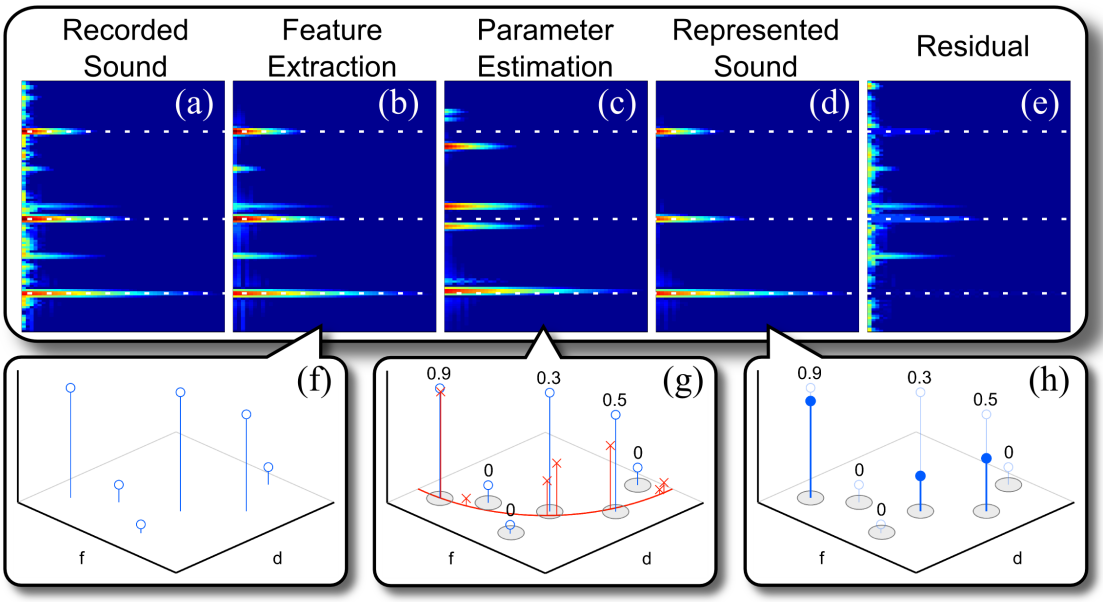


Figure 4.7: Residual computation. From a recorded sound (a), the reference features are extracted (b), with frequencies, dampings, and energies depicted as the blue circles in (f). After parameter estimation, the synthesized sound is generated (c), with the estimated features shown as the red crosses in (g), which all lie on a curve in the (f, d)-plane. Each reference feature may be approximated by one or more estimated features, and its match ratio number is shown. The represented sound is the summation of the reference features weighted by their match ratios, shown as the solid blue circles in (h). Finally, the difference between the recorded sound's power spectrogram (a) and the represented sound's (d) are computed to obtain the residual (e).

geometries and dynamics. Hence, we compute the transferred residual under the guidance of modes as follows.

Given a source geometry and impact point, we know how to transform its modal sound to a target geometry and impact points. Equivalently, we can describe such transformation as acting on the power spectrograms, transforming the modal power spectrogram of the source, $\mathbf{P}^s_{modal}$, to that of the target, $\mathbf{P}^t_{modal}$:

$$\mathbf{P}^s_{modal} \xrightarrow{H} \mathbf{P}^t_{modal} \tag{4.28}$$

where H is the transform function. We apply the same transform function H to the *residual* power spectrograms

$$\mathbf{P}^s_{residual} \xrightarrow{H} \mathbf{P}^t_{residual} \tag{4.29}$$

where the source residual power spectrogram is computed as described in Sec. 4.5.1.

More specifically, H can be decomposed into per-mode transform functions, $H_{i,j}$, which transforms the power spectrogram of a source mode $\phi_i^s = (f_i^s, d_i^s, a_i^s)$ to a target mode $\phi_j^t = (f_j^t, d_j^t, a_j^t)$. $H_{i,j}$ can further be described as a series of operations on the source power spectrogram $\mathbf{P}^s_{modal}$: (1) the center frequency is shifted from f_i^s to f_j^t; (2) the time dimension is stretched according to the ratio between d_i^s and d_j^t; (3) the height (intensity) is scaled pixel-by-pixel to match $\mathbf{P}^t_{modal}$. The per-mode transform is performed in the neighborhood of f_i^s, namely between $\frac{1}{2}(f_{i-1}^s + f_i^s)$ and $\frac{1}{2}(f_i^s + f_{i+1}^s)$, to that of f_j^t, namely between $\frac{1}{2}(f_{j-1}^t + f_j^t)$ and $\frac{1}{2}(f_j^t + f_{j+1}^t)$.

The per-mode transform is performed for all pairs of source and target modes, and the local residual power spectrograms are 'stitched' together to form the complete $\mathbf{P}^t_{residual}$. Finally, the time-domain signal of the residual is reconstructed from $\mathbf{P}^t_{residual}$, using an iterative inverse STFT algorithm by Griffin and Lim (2003). Algorithm 1 shows the complete feature-guided residual transfer algorithm.

Algorithm 1: Residual Transformation at Runtime

Input: *source* modes $\mathbf{\Phi}^s = \{\phi_i^s\}$, *target* modes $\mathbf{\Phi}^t = \{\phi_j^t\}$, and source residual audio $\mathbf{s}^s_{residual}[n]$
Output: target residual audio $\mathbf{s}^t_{residual}[n]$
$\mathbf{\Psi} \leftarrow$ DetermineModePairs($\mathbf{\Phi}^s, \mathbf{\Phi}^t$)
foreach *mode pair* $(\phi_k^s, \phi_k^t) \in \mathbf{\Psi}$ **do**
  $\mathbf{P}^{s\prime} \leftarrow$ ShiftSpectrogram($\mathbf{P}^s$, Δfrequency)
  $\mathbf{P}^{s\prime\prime} \leftarrow$ StretchSpectrogram($\mathbf{P}^{s\prime}$, damping_ratio)
  $\mathbf{A} \leftarrow$ FindPixelScale($\mathbf{P}^t$, $\mathbf{P}^{s\prime\prime}$)
  $\mathbf{P}^s_{residual}{}' \leftarrow$ ShiftSpectrogram($\mathbf{P}^s_{residual}$, Δfrequency)
  $\mathbf{P}^s_{residual}{}'' \leftarrow$ StretchSpectrogram($\mathbf{P}^s_{residual}{}'$, damping_ratio)
  $\mathbf{P}^t_{residual}{}'' \leftarrow$ MultiplyPixelScale($\mathbf{P}^s_{residual}{}''$, $\mathbf{A}$)
  $(\omega_{start}, \omega_{end}) \leftarrow$ FindFrequencyRange(ϕ_{k-1}^t, ϕ_k^t)
  $\mathbf{P}^t_{residual}$ [m, $\omega_{start}, \ldots, \omega_{end}$] $\leftarrow \mathbf{P}^t_{residual}{}''$ [m, $\omega_{start}, \ldots, \omega_{end}$]
end
$\mathbf{s}^t_{residual}[n] \leftarrow$ IterativeInverseSTFT($\mathbf{P}^t_{residual}$)

With this scheme, the transform of the residual power spectrogram is completely guided by the appropriate transform of modes. The resulting residual changes consistently with the modal sound. Since the modes transform with the geometry and dynamics in a physical manner, the transferred residual also faithfully reflects this variation.

Note that a 'one-to-one mapping' between the source and target modes is required. If the target geometry is a scaled version of the source geometry, then there is a natural correspondence between the modes. If the target geometry, however, is of different shape from the source one, such natural

correspondence does not exist. In this case, we pick the top $N_{dominant}$ modes with largest energies from both sides, and pair them from low frequency to high frequency.

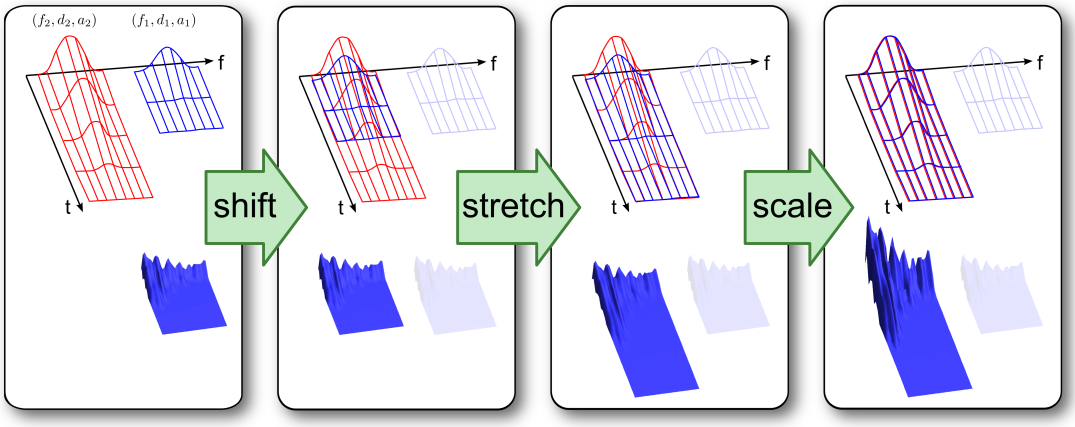


Figure 4.8: Single mode residual transform: The power spectrogram of a source mode (f_1, d_1, a_1) (the blue wireframe), is transformed to a target mode (f_2, d_2, a_2) (the red wireframe), through frequency-shifting, time-stretching, and height-scaling. The residual power spectrogram (the blue surface at the bottom) is transformed in the exact same way.

4.5.2.2 Implementation and Performance

The most computation costly part of residual transfer is the iterative inverse STFT process. We are able to obtain acceptable time-domain reconstruction from the power spectrogram when we limit the iteration of inverse STFT to 10. Hardware acceleration is used in our implementation to ensure fast STFT computation. More specifically, CUFFT, a CUDA implementation of Fast Fourier Transform, is adopted for parallelized inverse STFT operations. Also note that residual transfer computation only happens when there is a contact event, the obtained time-domain residual signal can be used until the next event. On an NVIDIA GTX 480 graphics card, if the contact events arrive at intervals around 1/30s, the residual transfer in the current implementation can be successfully evaluated in time.

4.5.2.3 Constants and Functions

We provide here the actual values and forms used in our implementation for the constants and functions introduced in Sec. 4.4.2,

For the relationship between critical-band rate z (in Bark) and frequency (in Hz), we use

$$Z(f) = 6\sinh^{-1}(f/600) \tag{4.30}$$

that approximates the empirically determined curve shown in Figure 4.4a (Wang et al., 1992).

We use $c_z = 5.0$ and $c_d = 100.0$ in Equation 4.21 and Equation 4.22.

In Equation 4.23, the weight w_i associated to a reference feature point ϕ_i is designed to be related to the energy of mode i. The energy can be found by integrating the power spectrogram of the damped sinusoid, and we made a modification such that the power spectrogram is transformed prior to integration. The image domain transformation introduced in Sec. 4.4.2.1, which better reflects the perceptual importance of a feature, is used.

The weight $\tilde{u}_{ij}$ used in Equation 4.24 is $\tilde{u}_{ij} = 0$ for $k(\phi_i, \tilde{\phi}_j) = 0$, and $\tilde{u}_{ij} = 1$ for $k(\phi_i, \tilde{\phi}_j) > 0$ (u_{ij} is defined similarly).

For the point-to-point match score $k(\phi_i, \tilde{\phi}_j)$ in Equation 4.24, we use

$$k(\phi_i, \tilde{\phi}_j) = k(D) = \begin{cases} 1.0 - 0.5D & \text{if } D \leq 1.0 \\ 0.5/D & \text{if } 1.0 < D \leq 5.0 \\ 0 & \text{if } 5.0 < D \end{cases} \tag{4.31}$$

where $D = D(\phi_i, \tilde{\phi}_j)$ is the Euclidean distance between the two feature points (Equation 4.20).

4.6 Results and Analysis

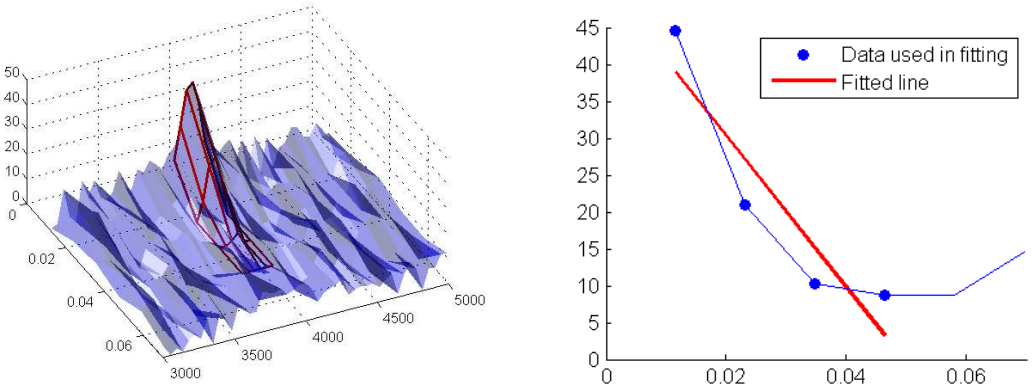
4.6.1 Feature Extraction

4.6.1.1 Comparison with Spectral Modeling Synthesis9

The Spectral Modeling Synthesis (SMS) method (Serra and Smith III, 1990) detects a peak also in the power spectrogram, tracks the one peak point over time, and forms an amplitude envelope. One can certainly use this amplitude envelope to infer the damping value, for example, by linear regression of the logarithmic amplitude values (which is the approach adopted by (Välimäki et al., 1996)). There are, however, several disadvantages of this approach.

First of all, tracking only the peak point over time implies that the frequency estimation is only accurate to the width of the frequency bins of power spectrogram. For example, for a window size of 512 samples, the width of a frequency bin is about 86 Hz, direct frequency peak tracking has frequency resolution as coarse as 86 Hz.

Serra and Smith pointed out this problem (Serra and Smith III, 1990), and proposes to improve the accuracy by taking the two neighboring frequency bins around the peak and performing a 3-point curve fitting to find the real peak (Serra, 1989). Our method takes a further step: instead of 3 points per time frame, we use all points within a rectangular region. The region extends as far as possible in both frequency and time axes until (a) the amplitude falls under a threshold to the peak amplitude, or (b) a local minimum in amplitude is reached. We then use an optimizer to find a damped sinusoid whose power spectrogram best matches the shape of the input data in the region of interest. An example is shown in Figure 4.9a, where the blue surface is the power spectrogram of the input sound clip, and the overlay red mesh is the power spectrogram of the best fitted damped sinusoid.

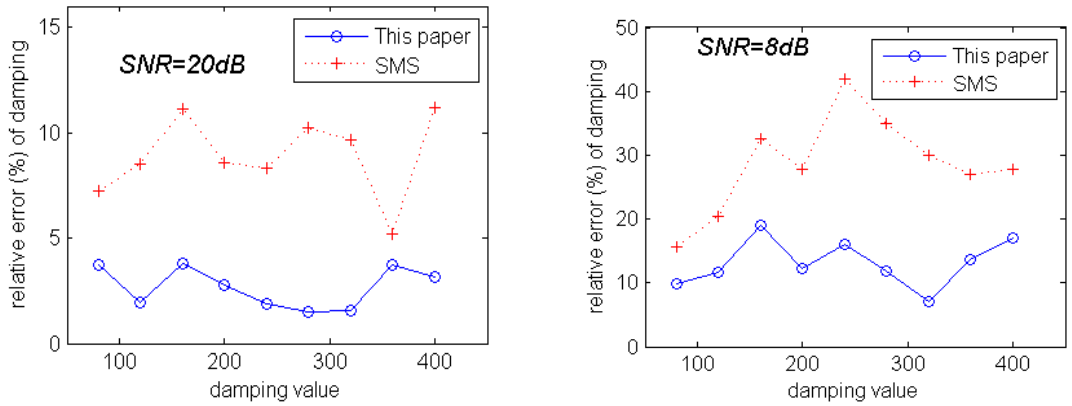


(a) (b)

Figure 4.9: Estimation of damping value in the presence of noise, using (a) our local shape fitting method and (b) SMS with linear regression.

Secondly, for linear regression to work well, there must be at least two points (the more the better) along the time axis, before the signal falls to the level of background noise. For high damping values, there will be only a few data points along the time axis. On the other hand, we know that the damping value is also reflected in the width of the hill, so when there are not enough points along the time axis, there are more points along the frequency axis with significant heights–which will help determining the damping value in our surface fitting method.

Taking more points into account makes it less sensitive to noise. In Figure 4.10, we simulated a noisy case where white noise with signal-to-noise ratio (SNR)=8 dB is added to a damped sinusoid with damping value 240, and use (a) our local surface fitting method and (b) SMS with linear regression to infer the damping value. In this particular example, due to the high damping value and high noise level, only 4 points participate in linear regression, while 24 points are considered in our method. Our shape fitting is less sensitive to irregularities than the fitted line in SMS. The average damping error versus damping value for both methods are plotted in Figure 4.10a and Figure 4.10b, where SNR=20 dB and 8 dB respectively.



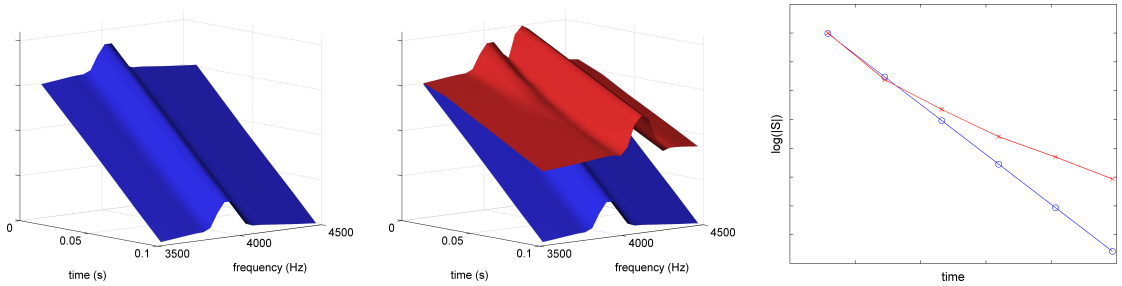
(a) (b)

Figure 4.10: Average damping error versus damping value for our method and SMS.

Mathematically, the 2D power spectrogram contains as much information as the original time domain signal (except for the windowing effect and the loss of phase). Using only a 1D sequence inevitably discards a portion of all available information (as in SMS), and in some cases (e.g. high damping values and high noise level) this portion is significant. Our surface matching method utilizes as much information as possible. Fitting a surface is indeed more costly than fitting a line, but it also achieves higher accuracy.

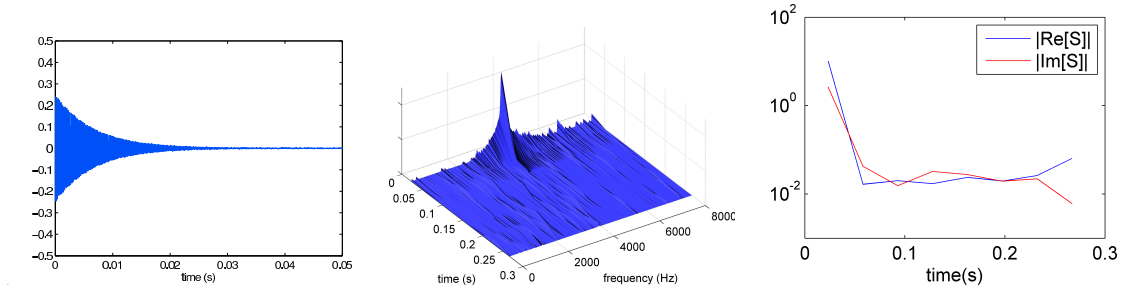
4.6.1.2 Comparison with a Phase Unwrapping Method

The 'phase unwrapping' technique proposed by (Pai et al., 2001) and (Corbett et al., 2007) is known for its ability to separate close modes within one frequency bin. Our method, however, works under a different assumption, and the ability to separate modes within a frequency bin has different



(a) (b) (c)

Figure 4.11: Interference from a neighboring mode located several bins away.



(a) (b) (c)

Figure 4.12: A noisy, high damping experiment.

impacts in our framework and theirs. In their framework, the extracted features $\{f_i, d_i, a_i\}$ are directly used in the sound synthesis stage and thus control the final audio quality. In our case, the features are only used to guide the subsequent parameter estimation process. In this process, two close modes will show up as near-duplicate points in the (f, d)-space. Because as pointed out by (Pai et al., 2001), modes with close frequencies usually result from the shape symmetry of the sounding object, and their damping values should also be close. In the process of fitting material parameters, or more specifically, in computing the feature domain metric, replacing these near-duplicate points with one point does not affect the quality of the result much.

Secondly, despite its ability to separate nearby modes, (Corbett et al., 2007) also proposes to merge modes if their difference in frequency is not greater than human's audible frequency discrimination limit (2-4 Hz). Among the multiple levels of power spectrograms that we used, the finest frequency resolution (about 3 Hz) is in fact around this limit.

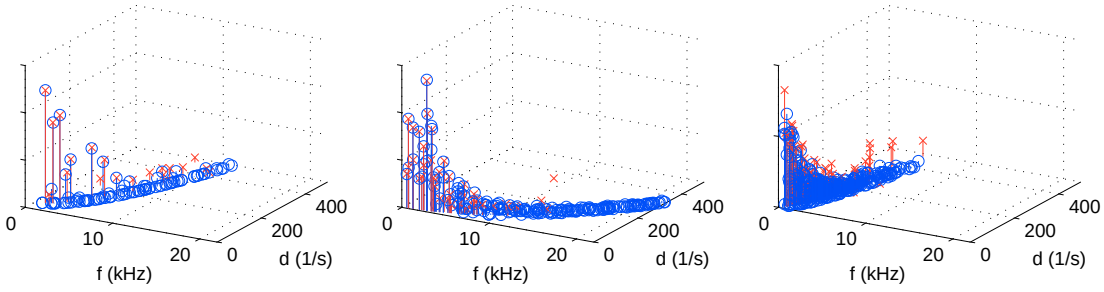
On the other hand, our proposed feature extraction algorithm offers some advantages and achieves higher accuracy compared with (Pai et al., 2001) and (Corbett et al., 2007) in some cases. When extracting the information of a mode, other modes within the same frequency bin

(which are successfully resolved by the Steiglitz-McBride algorithm (Steiglitz and McBride, 1965) underlying (Pai et al., 2001) and (Corbett et al., 2007)) are not the only source of interference. Other modes from several bins away also affect the values (complex or magnitude-only alike) in the current bin, known as the 'spillover effect'. In order to minimize this effect, the greedy method proposed in our work collects the modes with the largest average power spectral density first. Therefore, when examining a mode, the neighboring modes that have higher energy than the current one are already collected, and their influence removed. This can be demonstrated in Figure 4.11. The original power spectrogram of a mode (f_1, d_1, a_1) is shown in Figure 4.11a. The values at the frequency bin F_k containing f_1 are plotted over time, shown as the blue curve in Figure 4.11c. In Figure 4.11b, the the presence of another strong mode (f_2, d_2, a_2) located 5 bins away changes the values at F_k, plotted as the red curve in Figure 4.11c. The complex values of the STFT at F_k are not shown, but they are similarly interfered. If these complex values at F_k are directly fitted with the Steiglitz-McBride algorithm in the works by (Pai et al., 2001) and (Corbett et al., 2007), the estimated damping has a 20% error. The greedy approach in our multi-level algorithm removes the influence of the neighboring mode first, resulting in a 1% damping error.

Based on our experimentations, we also found that the universal frequency-time resolution used in (Pai et al., 2001) and (Corbett et al., 2007) is not always most suitable for all modes. Our method uses a dynamic selection of frequency-time resolution to address this problem. For example, in the case of high damping values, under a fixed frequency-time resolution, there may only be a few points above noise level along the time axis, which will undermine the accuracy of the Steiglitz-McBride algorithm. Figure 4.12 shows such an example, the damping value (150 s^{-1}) is high but not unreasonable, as shown in the time domain signal Figure 4.12a, where a white noise with SNR=60 dB is added. The power spectrogram is shown in Figure 4.12b. We implemented the method in the paper by (Corbett et al., 2007) using the suggested 46 ms window size (with $N_{overlap} = 4$) and tested on the above case. The input to this method is the complex values at the peak frequency bin, whose magnitudes of the real and imaginary parts are shown in Figure 4.12c, and an error of 5.7% for damping is obtained. As a comparison, our algorithm automatically selects a 23 ms window size and fits the local shape in a 6×5 region in the frequency-time space, yielding merely a 0.9% error for damping.

4.6.2 Parameter estimation

Before working on real-world recordings, we design an experiment to evaluate the effectiveness of our parameter estimation with synthetic sound clips. A virtual object with known material parameters $\{\alpha, \beta, \gamma, \sigma\}$ and geometry is struck, and a sound clip is synthesized by mixing the excited modes. The sound clip is entered to the parameter estimation pipeline to test if the same parameters are recovered. Three sets of parameters are tested and the results are shown in Figure4.13.



	truth	estimated	relative error
α	9.2003e+1	9.1995e+1	9.31e-5
β	1.8297e-7	1.8299e-7	9.30e-5
γ	3.6791e+0	3.6791e+0	3.91e-6
σ	2.1873e-3	2.1872e-3	5.61e-5

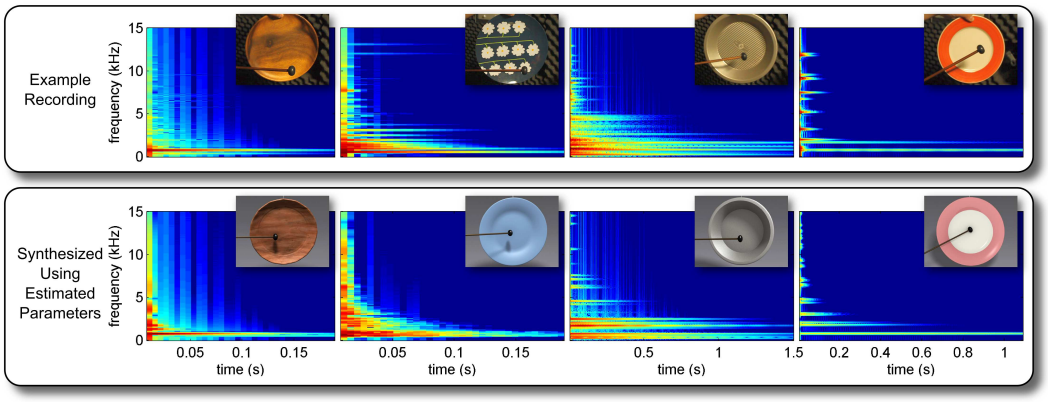
	truth	estimated	relative error
α	3.9074e+0	3.9069e+0	1.27e-4
β	3.3935e-8	3.3935e-8	1.62e-6
γ	3.4186e+0	3.4186e+0	1.17e-6
σ	9.0013e-6	9.0009e-6	4.67e-5

	truth	estimated	relative error
α	3.1425e+1	3.1428e+1	9.93e-5
β	7.0658e-7	7.0663e-7	7.61e-5
γ	7.3953e+0	7.3953e+0	3.00e-6
σ	3.5842e-9	3.5847e-9	1.46e-4

Figure 4.13: Results of estimating material parameters using synthetic sound clips. The intermediate results of the feature extraction step are visualized in the plots. Each blue circle represents a synthesized feature, whose coordinates (x, y, z) denote the frequency, damping, and energy of the mode. The red crosses represent the extracted features. The tables show the truth value, estimated value, and relative error for each of the parameters.

This experiment demonstrates that if the material follows the Rayleigh damping model, the proposed framework is capable of estimating the material parameters with high accuracy. Below we will see that real materials do not follow the Rayleigh damping model exactly, but the presented framework is still capable of finding the closest Rayleigh damping material that approximates the given material.

We estimate the material parameters from various real-world audio recordings: a wood plate, a plastic plate, a metal plate, a porcelain plate, and a glass bowl. For each recording, the parameters are estimated using a virtual object that is of the same size and shape as the one used to record the audio clips. When the virtual object is hit at the same location as the real-world object, it produces a sound similar to the recorded audio, as shown in Figure 4.14 and the supplementary video.



(a) *wood plate* (b) *plastic plate* (c) *metal plate* (d) *porcelain plate*

Figure 4.14: Parameter estimation for different materials. For each material, the material parameters are estimated using an example recorded audio (top row). Applying the estimated parameters to a virtual object with the same geometry as the real object used in recording the audio will produce a similar sound (bottom row).

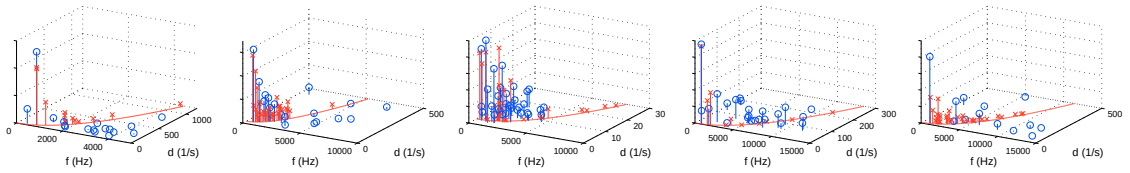
	Parameters			
Material	α	β	γ	σ
Wood	2.1364e+0	3.0828e-6	6.6625e+5	3.3276e-6
Plastic	5.2627e+1	8.7753e-7	8.9008e+4	2.2050e-6
Metal	6.3035e+0	2.1160e-8	4.5935e+5	9.2624e-6
Glass	1.8301e+1	1.4342e-7	2.0282e+5	1.1336e-6
Porcelain	3.7388e-2	8.4142e-8	3.7068e+5	4.3800e-7

Table 4.1: Refer to Sec. 4.1 and Sec. 4.4 for the definition and estimation of these parameters.

Figure 4.15 compares the refenece features of the real-world objects and the estimated features of the virtual objects as a result of the parameter estimation. The parameter estimated for these materials are shown in Table. 4.1.

Transfered parameters and residual: The parameters estimated can be transfered to virtual objects with different sizes and shapes. Using these material parameters, a different set of resonance modes can be computed for each of these different objects. The sound synthesized with these modes preserves the intrinsic material quality of the example recording, while naturally reflect the variation in virtual object's size, shape, and interactions in the virtual environment.

Moreover, taking the difference between the recording of the example real object and the synthesized sound from its virtual counterpart, the residual is computed. This residual can also be transfered to other virtual objects, using methods described in Sec. 4.5.



(a) wood plate (b) plastic plate (c) metal plate (d) porcelain (e) glass bowl

Figure 4.15: Feature comparison of real and virtual objects. The blue circles represent the reference features extracted from the recordings of the real objects. The red crosses are the features of the virtual objects using the estimated parameters. Because of the Rayleigh damping model, all the features of a virtual object lie on the depicted red curve on the (f, d)-plane.

Figure 4.16 gives an example of this transferring process. From an example recording of a porcelain plate (a), the parameters for the porcelain material are estimated, and the residual computed (b). The parameters and residual are then transfered to a smaller porcelain plate (c) and a porcelain bunny (d).

4.6.3 Comparison with real recordings

Figure 4.17 shows a comparison of the transferred results with the real recordings. From a recording of glass bowl, the parameters for glass are estimated (column (a)) and transfered to other virtual glass bowls of different sizes. The synthesized sounds ((b) (c) (d), bottom row) are compared with the real-world audio for these different-sized glass bowls ((b) (c) (d), top row). It can be seen that although the transfered sounds are not identical to the recorded ones, the overall trend in variation is similar. Moreover, the perception of material is preserved, as can be verified in the accompanying video. More examples of transferring the material parameters as well as the residuals are demonstrated in the accompanying video.

4.6.4 Example: a complicated scenario

We applied the estimated parameters for various virtual objects in a scenario where complex interactions take place, as shown in Figure 4.18 and the accompanying video.

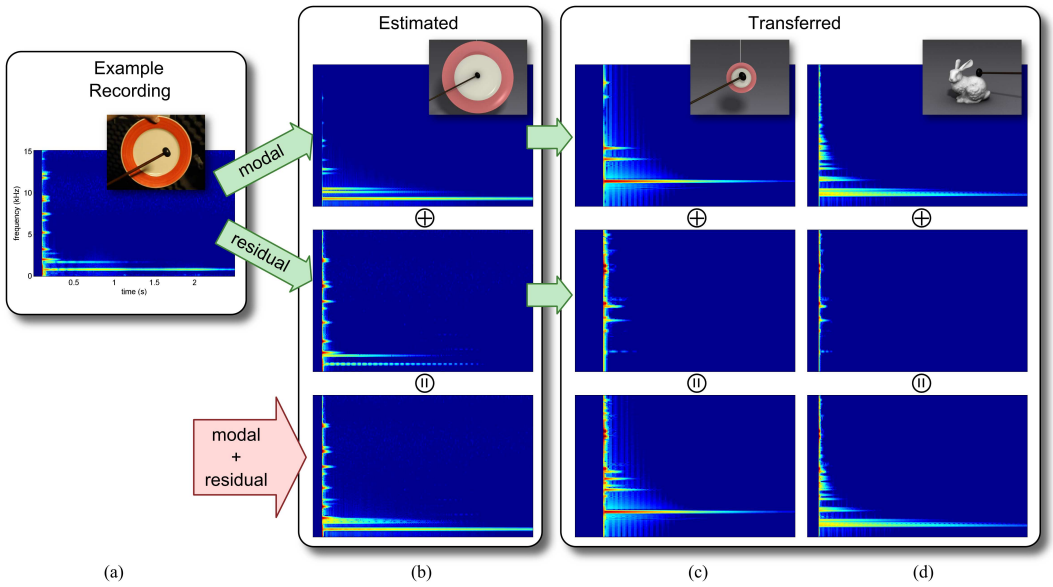


Figure 4.16: Transfered material parameters and residual: from a real-world recording (a), the material parameters are estimated and the residual computed (b). The parameters and residual can then be applied to various objects made of the same material, including (c) a smaller object with similar shape; (d) an object with different geometry. The transfered modes and residuals are combined to form the final results (bottom row).

4.6.5 Performance

Table 4.2 shows the timing for our system running on a single core of a 2.80 GHz Intel Xeon X5560 machine. It should be noted that the parameter estimation is an offline process: it needs to be run only once per material, and the result can be stored in a database for future reuse.

For each material in column one, multiple starting points are generated first as described in Sec. 4.4.3, and the numbers of starting points are shown in column two. From each of these starting points, the optimization process runs for an average number of iterations (column three) until convergence. The average time taken for the process to converge is shown in column four. The convergence is defined as when both the step size and the difference in metric value are lower than their respective tolerance values, Δ_x and Δ_{metric}. The numbers reported in Table 4.2 are measured with Δ_x = 1e-4 and Δ_{metric} = 1e-8.

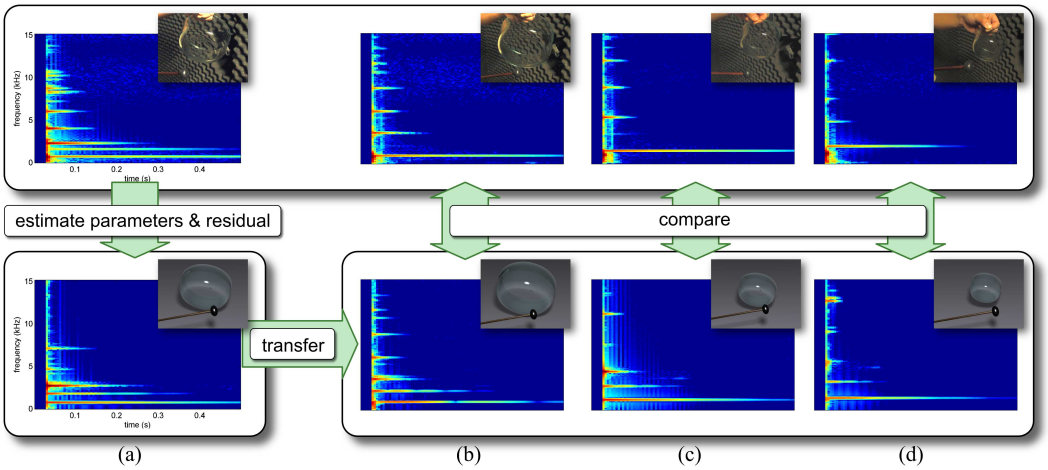


Figure 4.17: Comparison of transfered results with real-word recordings: from one recording (column (a), top), the optimal parameters and residual are estimated, and a similar sound is reproduced (column (a), bottom). The parameters and residual can then be applied to different objects of the same material ((b), (c), (d), bottom), and the results are comparable to the real-world recordings ((b), (c), (d), top).

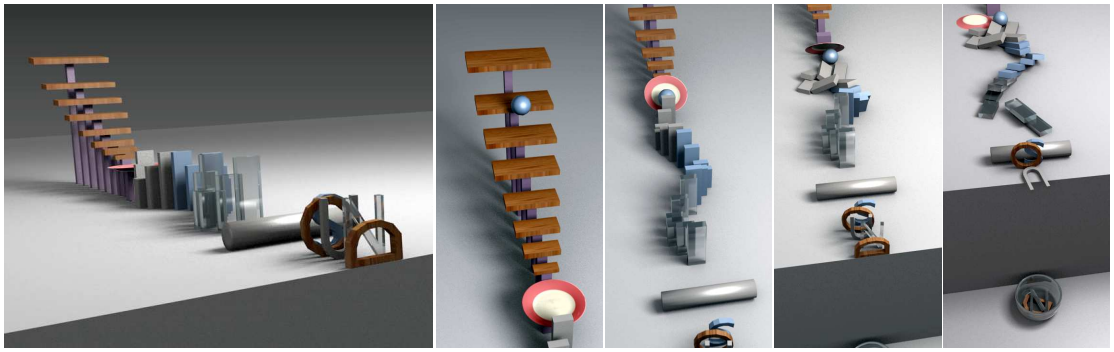


Figure 4.18: The estimated parameters are applied to virtual objects of various sizes and shapes, generating sounds corresponding to all kinds of interactions such as colliding, rolling, and sliding.

4.7 Conclusion and Future Work

We have presented a novel data-driven, physically based sound synthesis algorithm using an example audio clip from real-world recordings. By exploiting psychoacoustic principles and feature identification using linear modal analysis, we are able to estimate the appropriate material parameters that capture the intrinsic audio properties of the original materials and transfer them to virtual objects of different sizes, shape, geometry and pair-wise interaction. We also propose an effective residual computation technique to compensate for linear approximation of modal synthesis.

Although our experiments show successful results in estimating the material parameters and computing the residuals, it has some limitations. Our model assumes linear deformation and Rayleigh

Material	#starting points	average #iteration	average time (s)
Wood	60	1011	46.5
Plastic	210	904	49.4
Metal	50	1679	393.5
Porcelain	80	1451	131.3
Glass	190	1156	68.9

Table 4.2: **Offline Computation for Material Parameter Estimation**

damping. While offering computational efficiency, these models cannot always capture all sound phenomena that real world materials demonstrate. Therefore, it is practically impossible for the modal synthesis sounds generated with our estimated material parameters to sound exactly the same as the real-world recording. Our feature extraction and parameter estimation depend on the assumption that the modes do not couple with one another. Although it holds for the objects in our experiments, it may fail when recording from objects of other shapes, e.g. thin shells where nonliear models would be more appropriate (Chadwick et al., 2009).

We also assume that the recorded material is homogeneous and isotropic. For example, wood is highly anisotropic when measured along or across the direction of growth. The anisotropy greatly affects the sound quality and is an important factor in making high-precision musical instruments.

Because the sound of an object depends both on its geometry and material parameters, the geometry of the virtual object must be as close to the real-world object as possible to reduce the error in parameter estimation. Moreover, the mesh discretization must also be adequately fine. For example, although a cube can be represented by as few as eight vertices, a discretization so coarse not only clips the number of vibration modes but also makes the virtual object artificially stiffer than its real-world counterpart. The estimated γ, which encodes the stiffness, is thus unreliable. These requirements regarding the geometry of the virtual object may affect the accuracy of the results using this method.

Although our system is able to work with an inexpensive and simple setup, care must be taken in the recording condition to reduce error. For example, the damping behavior of a real-world object is influenced by the way it is supported during recording, as energy can be transmitted to the supporting device. In practice, one can try to minimize the effect of contacts and approximate the system as free vibration, or one can rigidly fix some points of the object to a relatively immobile structure and

model the fixed points as part of the boundary conditions in the modal analysis process. It is also important to consider the effect of room acoustics. For example, a strong reverberation will alter the observed amplitude-time relationship of a signal and interfere with the damping estimation.

Despite these limitations, our proposed framework is general, allowing future research to further improve and use different individual components. For example, the difference metric now considers the psychoacoustic factors and material resemblance through power spectrogram comparison and feature matching. It is possible that more factors can be taken into account, or a more suitable representation, as well as a different similarity measurement of sounds can be found.

The optimization process approximates the global optimum by searching through all 'good' starting points. With a deeper investigation of the parameter space and more experiments, the performance may be possibly improved by designing a more efficient scheme to navigate the parameter space, such as starting-point clustering, early pruning, or a different optimization procedure can be adopted.

Our residual computation compensates the difference between the real recording and the synthesized sound, and we proposed a method to transfer it to different objects. However, it is not the only way – much due to the fact that the origin and nature of residual is unknown. Meanwhile, it still remains a challenge to acquire recordings of only the stuck object and completely remove input from the striker. Our computed residual is inevitably polluted by the striker to some extent. Therefore, future solutions for separating sounds from the two interacting objects should facilitate a more accurate computation for residuals from the struck object.

When transferring residual computed from impacts to continuous contacts (e.g. sliding and rolling), there are certain issues to be considered. Several previous work have approximated continuous contacts with a series of impacts and have generated plausible *modal* sounds. Under this approximation, our proposed feature-guided residual transfer technique can be readily adopted. However, the effectiveness of this direct mapping needs further evaluation. Moreover, future study on continuous contact sound may lead to an improved modal synthesis model different than the impact-based approximation, under which our residual transfer may not be applicable. It is then also necessary to reconsider how to compensate the difference between a real continuous contact sound and the modal synthesis sound.

In this chapter, we focus on designing a system that can quickly estimate the optimal material parameters and compute the residual merely based on a *single* recording. However, when a small number of recordings of the same material are given as input, machine learning techniques can be used to determine the set of parameters with maximum likelihood, and it could be an area worth exploring. Finally, we would like to extend this framework to other non-rigid objects and fluids, and possibly nonlinear modal synthesis models as well.

In summary, data-driven approaches have proven useful in areas in computer graphics, including rendering, lighting, character animation, and dynamics simulation. With promising results that are transferable to virtual objects of different geometry, sizes, and interactions, this work is the first rigorous treatment of the problem on automatically determining the material parameters for physically based sound synthesis using a single sound recording, and it offers a new direction for combining example-guided and modal-based approaches.

CHAPTER 5: WAVE-RAY HYBRID SOUND PROPAGATION

The previous chapters focused on *sound synthesis* techniques that I have developed for liquid sounds and rigid body sounds. The aim of this chapter is to describe a technique that I have developed for *sound propagation*, which is a hybird technique combining wave simulation and ray-tracing based acoustic techniques. The chapter is organized as follows: first I give an overview to our hybrid sound propagation technique, followed by an in-depth discussion of the key component, the tw-way coupling procedure. Then I describe the implementation of the sound propagation system, the results obtained from it, and the performance and error analysis. Finally, I conclude with a summary of my contribution and a discussion of possible future work.

5.1 Overview

In this section we give an overview of sound propagation and our proposed approach.

5.1.1 Sound Propagation

For a sound pressure wave with angular frequency ω, speed of sound c, the problem of sound propagation in domain Ω in the space can be expressed as a boundary value problem for the Helmholtz equation :

$$\nabla^2 p + \frac{\omega^2}{c^2} p = f; \quad \mathbf{x} \in \Omega, \tag{5.1}$$

where $p(\mathbf{x})$ is the complex valued pressure field, ∇^2 is the Laplacian operator, and $f(\mathbf{x})$ is the *source term*, (e.g. $= 0$ in free space and $\delta(\mathbf{x}')$ for a point source located at $\mathbf{x}'$). Boundary conditions are specified on the boundary $\partial\Omega$ of the domain (which can be the surface of an solid object, the interface between different media, or an arbitrarily defined surface) by a Dirichlet boundary condition that specifies pressure, $p(\mathbf{x}) = 0; \mathbf{x} \in \partial\Omega$, a Neumann boundary condition that specifies the velocity of medium, $\frac{\partial p(\mathbf{x})}{\partial n} = 0; \mathbf{x} \in \partial\Omega$, or a mixed boundary condition that specifies a complex-valued constant Z, so that $Z\frac{\partial p(\mathbf{x})}{\partial n} + p(\mathbf{x}) = 0; \mathbf{x} \in \partial\Omega$.

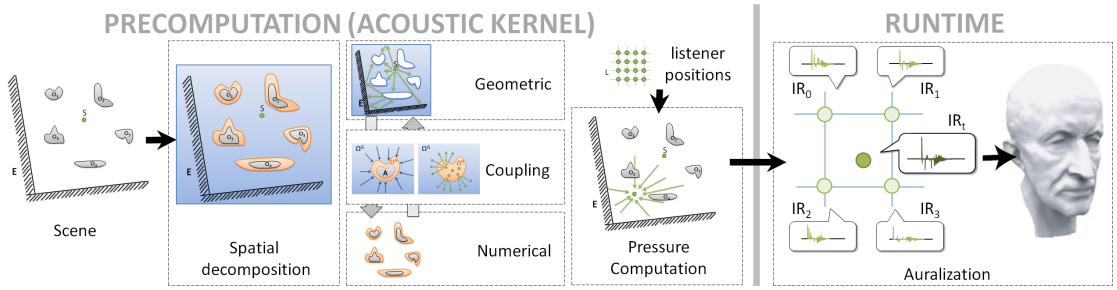


Figure 5.1: Overview of spatial decomposition in our hybrid sound propagation technique: In the precomputation phase, a scene is classified into objects and environment features. This includes near-object regions (shown in orange) and far-field regions (shown in blue). The sound field in near-object regions is computed using a numerical wave simulation, while the sound field in far-field region is computed using geometric acoustic techniques. A two-way coupling procedure couples the results computed by geometric and numerical methods. The sound pressures are computed at different listener positions to generate the impulse responses. At runtime, the precomputed impulse responses (IR_0-IR_3) are retrieved and interpolated for the specific listener position (IR_t) at interactive rates, and final sound is rendered.

The pressure p at infinity must also be specified, usually by the *Sommerfeld radiation condition* (Pierce, 1989), $\lim_{\|\mathbf{x}\|\to\infty}\left[\frac{\partial p}{\partial \|\mathbf{x}\|} + \hat{j}\omega c p\right] = 0$, where $\|\mathbf{x}\|$ is the distance of point $\mathbf{x}$ from the origin and $\hat{j} = \sqrt{-1}$.

Different acoustic techniques aim to solve the above equations with different formulations. Numerical acoustic techniques discretize Equation (5.1) and solve for p numerically with boundary conditions. Geometric acoustic techniques model p as a discrete set of rays emitted from sound sources which interact with the environment and propagate the pressure.

5.1.2 Acoustic Transfer Function

When modeling the acoustic effects due to objects or surfaces in a scene, it is often useful to define the *acoustic transfer function*. Many different acoustic transfer functions have been proposed to simulate different acoustic effects. In sound propagation problems, the acoustic transfer function maps an incoming sound field to an outgoing sound field. For example, Waterman developed a *transition-matrix method* for acoustic scattering (Waterman, 2009) and maps the incoming and outgoing fields in terms of the coefficients of a complete system of vector basis functions. Antani et al. (2012) compute an acoustic radiance transfer operator that maps incident sound to diffusely reflected sound in a scene. Mehra et al. (2013) model the free-field acoustic behavior of an object, as

well as pairwise interactions between objects. In sound radiation problems, James et al. (2006b) map the vibration mode of an object to the radiated sound pressure field.

5.1.3 Hybrid Sound Propagation

We describe the various components of our hybrid sound propagation technique. Our approach uses a combination of frequency decomposition and spatial decomposition, as shown schematically in Figure 5.2. Since frequency decomposition is a standard technique (Granier et al., 1996), we mostly focus on spatial decomposition and our novel two-way coupling algorithm (see Figure 5.1).

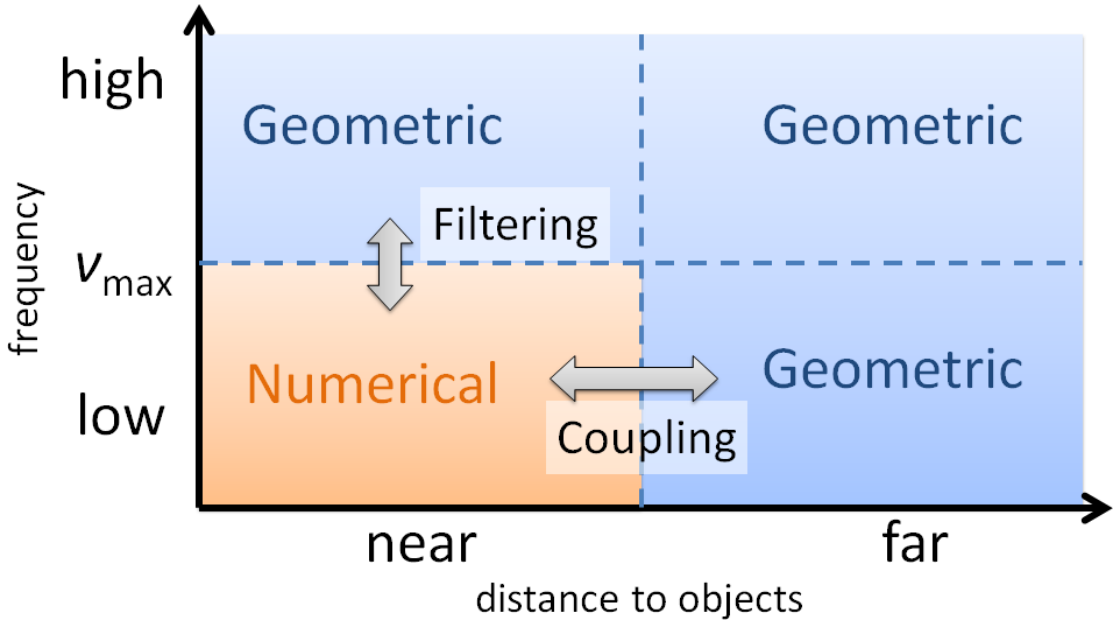


Figure 5.2: Frequency and spatial decomposition. High frequencies are simulated using geometric techniques, while low frequencies are simulated using a combination of numerical and geometric techniques based on a spatial decomposition.

Frequency Decomposition: We divide the modeled frequencies to low and high frequencies, with a crossover frequency ν_{max}. For high frequencies, geometric techniques are used throughout the entire domain. For low frequencies, a combination of numerical and geometric techniques is used based on a spatial decomposition described below. Typical values for ν_{max} are 0.5-2 kHz, and a simple low-pass–high-pass filter combination is usually used to join the results at the crossover frequency region.

Spatial decomposition: Given a scene we first classify it into *small objects* and *environment features*. The small objects, or simply *objects*, are of size comparable to or smaller than the wavelength of the

sound pressure wave being simulated. The environment features represent objects much larger than the wavelength (like terrain). The wavelength that is used as the criterion for distinguishing small or large objects is a user-controlled parameter. One possible choice is the maximum audible wavelength (17 m), corresponding to the lowest audible frequency for human (20 Hz). When sound interacts with objects, wave phenomena are prominent only when the objects are small relative to the wavelength. Therefore we only need to compute accurate wave propagation in the local neighborhood of small objects. We call this neighborhood the *near-object region* (orange region in Figure 5.1) of an object, and numerical acoustic techniques are used to compute the sound pressure field in this region. The region of space away from small objects is called the *far-field region* and is handled by geometric acoustic techniques (blue region in Figure 5.1).

The spatial decomposition is performed as follows: For a small object A, we compute the offset surface ∂A^+ and define the near-object region, denoted as Ω^N, as the space inside the offset surface. The offset surface of an object is computed using discretized distance fields and the marching cubes algorithm similar to James et al. (2006b). If the offset surfaces of two objects intersect then they are treated as a single object and are enclosed in one Ω^N. The space complementary to the near-object region is defined as the far-field region, and is denoted as Ω^G.

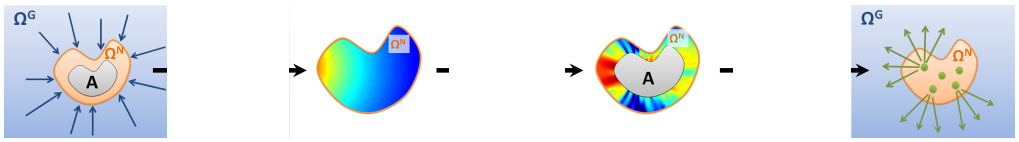
Geometric acoustics: The pressure waves constituting the sound field in Ω^G are modeled as a discrete set of rays. Their propagation in space and interaction with environment features (e.g. reflection from walls) are governed by geometric acoustic principles. We denote the pressure value defined collectively by the rays at position $\mathbf{x}$ as $p^G(\mathbf{x})$,

$$p^G(\mathbf{x}) = \sum_{r \in R} p_r(\mathbf{x}), \tag{5.2}$$

where p_r is the contribution from one ray r in a set of rays R.

Numerical acoustic techniques: The sound pressure field scattered by objects in Ω^N is treated by wave-based numerical techniques for lower frequencies, in which the wave phenomena such as diffraction and interference are inherently modeled. We denote the pressure value at position $\mathbf{x}$ computed using numerical techniques as $p^N(\mathbf{x})$.

Coupling: At the interface between near-object and far-field regions, the pressures computed by the two different acoustic techniques need to be coupled (Figure 5.3). Rays entering a near-object



(a) $\Omega^G \rightarrow \Omega^N$ (b) p_{inc} (c) p_{sca} (d) $\Omega^N \rightarrow \Omega^G$

Figure 5.3: Two-way coupling of pressure values computed by geometric and numerical acoustic techniques. (a) The rays are collected at the boundary and the pressure evaluated. (b) The pressure on the boundary defines the incident pressure field p_{inc} in Ω^N, which serves as the input to the numerical solver. (c) The numerical solver computes the scattered field p_{sca}, which is the effect of object A to the pressure field. (d) p_{sca} is expressed as fundamental solutions and represented as rays emitted to Ω^G.

region define the incident pressure field that serves as the input to the numerical solver. Similarly, the outgoing scattered pressure field computed by the numerical solver must be converted to a set of rays. The two-way coupling are modeled as transfer functions between incoming and outgoing rays. The process is detailed in Section 5.2.

Pressure computation: At each frequency lower than ν_{max}, the coupled geometric and numerical methods are used to solve the global sound pressure field. All frequencies higher than ν_{max} are handled by geometric techniques throughout the entire domain.

Acoustic kernel: The previous stages serve as an *acoustic kernel*, which computes the impulse responses (IRs) for a given source-listener position pair. For each sound source, the pressure value at each listener position is evaluated for all simulated frequencies to give a complete acoustic *frequency response* (FR), which can in turn be converted to an impulse response (IR) through Fourier transform. IR's for predefined source-listener positions (usually on a grid) are precomputed and stored.

Auralization: At runtime, the IR for a general listener position is obtained by interpolating the neighboring precomputed IR's (Raghuvanshi et al., 2010), and the output sound is auralized by convoluting the input sound with the IRs in real time.

5.2 Two-Way Wave-Ray Coupling

In this section, we present the details of our two-way coupling procedure. We also highlight the precomputation and runtime phases. The coupling procedure ensures the consistency between p^G and p^N, the pressures computed by the geometric and numerical acoustic techniques, respectively.

Any exchange of information at the interface between Ω^G and Ω^N must result in valid solutions to the Helmholtz equation (5.1) in both domains Ω^G and Ω^N.

5.2.1 Geometric → Numerical

From the pressure field p^G, we want to find the *incident pressure field* p_{inc}, which serves as the input to the numerical solver inside Ω^N. The incident pressure field is defined as the pressure field that corresponds to the solution of the wave equation if there were no objects in Ω^N.

Mathematically p_{inc} is the solution of the free-space Helmholtz Equation (5.1) with forcing term $f = 0$. Since there is no object in domain Ω^G,

$$p_{\text{inc}}(\mathbf{x}) = p^G(\mathbf{x}); \quad \mathbf{x} \in \Omega^G. \tag{5.3}$$

This equation defines a Dirichlet boundary condition on the interface ∂A^+:

$$p = p^G(\mathbf{x}); \quad \mathbf{x} \in \partial A^+, \tag{5.4}$$

The uniqueness of the acoustic boundary value problem guarantees that the solution of the free-space Helmholtz Equation, along with the specified boundary condition, is unique inside Ω^N. The unique solution $p_{\text{inc}}(\mathbf{x})$ can be found by expressing it as a linear combination of *fundamental solutions*. [1] If $\varphi_i(\mathbf{x})$ is a fundamental solution, and $p_{\text{inc}}(\mathbf{x})$ is expressed as a linear combination,

$$p_{\text{inc}}(\mathbf{x}) = \sum_i c_i \varphi_i(\mathbf{x}) \quad \mathbf{x} \in \Omega^N, \tag{5.5}$$

then the linearity of the wave equation implies that $p_{\text{inc}}(\mathbf{x})$ is also a solution. Furthermore, if the coefficients c_i are such that the boundary condition (5.4) is satisfied, then $p_{\text{inc}}(\mathbf{x})$ is the required *unique* solution to the boundary value problem (Section 3 in Ochmann (1995)). Therefore, the resultant pressure field is a valid incoming field in the numerical domain. The numerical solver takes the incident pressure field, considers the effect of the object inside Ω^N, and computes the outgoing scattered field. Figures 5.3(a) and 5.3(b) illustrate the process.

[1] A fundamental solution F for a linear operator L (in this case the Helmholtz operator $L = \nabla^2 + \frac{\omega^2}{c^2}$) is defined as the solution to the equation $LF = \delta(\mathbf{x})$, where δ is the Dirac delta function (Vladimirov, 1976).

5.2.2 Numerical → Geometric

In order to transfer information from Ω^N to Ω^G, a discrete set of rays must be determined to represent the computed pressure p^N. These outgoing rays may be emitted from some starting points located in Ω^N and carry different information related to the modeled pressure waves (strength, phase, frequency, spatial derivatives of pressure, etc.) The coupling procedure thus needs to compute the appropriate outgoing rays, given the numerically computed p^N.

The scattered field in the numerical domain due to the object can be simply written as,

$$p_{\text{sca}}(\mathbf{x}) = p^N(\mathbf{x}); \quad \mathbf{x} \in \Omega^N. \tag{5.6}$$

We need to find the scattered field outside of Ω^N, and model it as a set of rays. As before, Equation (5.6) defines a Dirichlet boundary condition on the interface ∂A^+,

$$p = p^N(\mathbf{x}); \quad \mathbf{x} \in \partial A^+. \tag{5.7}$$

The free space Helmholtz Equation, along with this boundary condition, uniquely defines the scattered field p_{sca} outside Ω^N. We again express p_{sca} as a linear combination of fundamental solutions φ_j:

$$p_{\text{sca}}(\mathbf{x}) = \sum_j c_j \varphi_j(\mathbf{x}); \quad \mathbf{x} \in \Omega^G, \tag{5.8}$$

and then find the coefficients c_j by satisfying the boundary condition (5.7). This gives us a unique solution for scattered field $p_{\text{sca}}(\mathbf{x})$ outside Ω^N. We then use a set of rays R_j^{out} to model the fundamental solutions $\varphi_j(\mathbf{x})$ such that

$$\varphi_j(\mathbf{x}) = \sum_{r \in R_j^{\text{out}}} p_r(\mathbf{x}), \quad \mathbf{x} \in \Omega^G. \tag{5.9}$$

These rays correctly represent the outgoing scattered field in Ω^G. Figure 5.3(c) and 5.3(d) illustrate the process.

The coupling process described above is a general formulation and is independent of the underlying numerical solver (BEM, FEM, etc.) that is used to compute p^N as long as the pressure on the interface ∂A^+ can be evaluated and expressed as a set of fundamental solutions. Depending

on the mathematical formulation of the selected set of fundamental solutions $\varphi_j(\mathbf{x})$, different rays (starting points, directions, information carried, etc.) can be defined. However, a general principle is that if $\varphi_j(\mathbf{x})$ has a singularity at $\mathbf{y}_j$, then $\mathbf{y}_j$ is a natural starting point from which rays are emitted. The directions of rays sample a unit sphere uniformly or with some distribution function (e.g. guided sampling (Taylor et al., 2012)). The choice of fundamental solutions will be discussed in the next section.

Note that if the fundamental solutions φ_i and φ_j used to express the incident field (Equation (5.5)) and outgoing scattered field (Equation (5.8)) are *predetermined*, then the mapping from φ_i to φ_j can be precomputed. This precomputation process will be discussed in section 5.2.4.

5.2.3 Fundamental solutions

The requirement for the choice of fundamental solution φ_j is that it must satisfy the Helmholtz Equation (5.1) and the Sommerfeld radiation condition.

Equivalent Sources: One choice of fundamental solutions is based on *equivalent sources* (Ochmann, 1995). Each fundamental solution is chosen to correspond to the field due to *multipole sources* of order L ($L = 1$ is a monopole, $L = 2$ is a dipole, etc.) located at $\mathbf{y}_j$:

$$\varphi_j(\mathbf{x}) = \varphi_{jlm}(\mathbf{x}), \tag{5.10}$$

for $l \leq L - 1$ and $-l \leq m \leq l$, and

$$\varphi_{jlm} = \Gamma_{lm} h_l^{(2)}(\omega\rho_j/c)\psi_{lm}(\theta_j, \phi_j), \tag{5.11}$$

where $(\rho_j, \theta_j, \phi_j)$ corresponds to the vector $(\mathbf{x} - \mathbf{y}_j)$ expressed in spherical coordinates, $h_l^{(2)}(\cdot)$ is the complex-valued spherical Hankel function of the second kind, $\psi_{lm}(\theta_j, \phi_j)$ is the complex-valued spherical harmonic function, and Γ_{lm} is the real-valued normalizing factor that makes the spherical harmonics orthonormal (Arfken et al., 1985). We use a shorthand generalized index h for (l, m), such that $\varphi_{jh}(\mathbf{x}) \equiv \varphi_{jlm}(\mathbf{x})$.

For pressure fields outside of ∂A^+ (i.e. in Ω^G), these equivalent sources are placed inside of ∂A^+ (i.e. in Ω^N). In a similar fashion, for pressure fields inside Ω^N, the equivalent sources must be placed outside Ω^N.

We model the outgoing pressure field from these equivalent sources using rays (Equation (5.9)) as follows. Rays are emitted from the source location $\mathbf{y}_j$. For a ray of direction (θ, ϕ) that has traveled a distance ρ, its pressure is scaled by $\psi_{lm}(\theta, \phi)$ and $h_l^{(2)}(\omega\rho/c)$.

Note that we can use equivalent sources to express a pressure field independently of how the pressure field was computed. For a computed p^N, we only need to find the locations $\mathbf{y}_j$ and coefficients c_j of the equivalent sources. This is performed by satisfying the boundary condition (5.8) in a least squared sense.

Boundary Elements: If the underlying numerical acoustic technique of choice is the boundary element method (BEM), then another set of fundamental solutions which is directly based on the BEM formulation is possible. For a domain with boundary $\partial\Omega$, the boundary element method solves the boundary integral equation of the Helmholtz equation. The boundary $\partial\Omega$ is discretized into triangular surface elements, and the equation is solved numerically for two variables; the pressure p and its normal derivative $\frac{\partial p}{\partial n}$ on the boundary. Once the boundary solutions p and $\frac{\partial p}{\partial n}$ are known, the sound pressure in the domain can be found for any point $\mathbf{x}$ by summing all the contributions from the surface triangles:

$$p(\mathbf{x}) = \int_{\partial\Omega} \left(G(\mathbf{y}, \mathbf{x}) \frac{\partial p(\mathbf{y})}{\partial n} - \frac{\partial G(\mathbf{y}, \mathbf{x})}{\partial n} p(\mathbf{y}) \right) d\left(\partial\Omega(\mathbf{y})\right), \tag{5.12}$$

where $\mathbf{y}$ is the approximated position of the triangle and G is the Green's Function $G(\mathbf{y}, \mathbf{x}) = \exp(\hat{j}\omega|\mathbf{x} - \mathbf{y}|/c)/4\pi|\mathbf{x} - \mathbf{y}|$ (Gumerov and Duraiswami, 2009).

Note that the discretization of Equation (5.12) also takes the form of Equation (5.8) as a linear combination of fundamental solutions:

$$p(\mathbf{x}) = \sum_j \left(c_j^1 \varphi_j^1(\mathbf{x}) + c_j^2 \varphi_j^2(\mathbf{x}) \right), \tag{5.13}$$

where the two kinds of fundamental solutions are

$$\varphi_j^1(\mathbf{x}) = G(\mathbf{y_j}, \mathbf{x}) \frac{\partial p(\mathbf{y_j})}{\partial n}; \; \varphi_j^2(\mathbf{x}) = -\frac{\partial G(\mathbf{y_j}, \mathbf{x})}{\partial n} p(\mathbf{y_j}). \tag{5.14}$$

Under this formulation, we can represent the pressure field as two kinds of rays emitted from each triangle location $\mathbf{y_j}$, each modeling $\varphi_j^1(\mathbf{x})$ and $\varphi_j^2(\mathbf{x})$ respectively. Then for a point in Ω^G the pressure field defined by the rays is computed according to Equation (5.12).

5.2.4 Precomputed Transfer Functions

If we consider what happens in Ω^N as a black box, the net result of the coupling and the numerical solver is that a set of rays enter Ω^N and then another set of rays exit Ω^N:

$$R^{\text{in}} \xrightarrow{\mathcal{M}} R^{\text{out}}, \tag{5.15}$$

where R^{in} is the set of incoming rays entering Ω^N, R^{out} is the set of outgoing rays, and $\mathcal{M}$ is the ray transfer function. In this case, the function $\mathcal{M}$ is similar to the bidirectional reflectance distribution function (BRDF) for light (Ben-Artzi et al., 2008). In our formulation, $\mathcal{M}$ encodes all the operations for the following computations:

1. Collect pressures defined by R^{in} to form the incident field on the interface (Equation (5.4));
2. Express the incident field as a set of fundamental solutions (Equation (5.5));
3. Compute the outgoing scattered field using the numerical acoustic technique;
4. Express the outgoing scattered field as a set of fundamental solutions (Equation (5.8); and finally,
5. Find a set of rays R^{out} that model these functions (Equation (5.9).

A straightforward realization of hybrid sound propagation technique is possible: from each sound source rays are traced, interacting with the environment features, entering and exiting the near-object regions transfered by different $\mathcal{M}$'s, and finally reaching a listener. However, as the first step of $\mathcal{M}$ depends on the incoming rays R^{in}, a different $\mathcal{M}$ must be computed each time the rays enter the same near-object region. Moreover, the process must be repeated until the solution

converges to a steady state, which may be too time-consuming for a scene (e.g. an indoor scene) with multiple ray reflections causing multiple entrances to near-object regions.

While previous two-way hybrid techniques do not consider this problem (Barbone et al., 1998; Jean et al., 2008), we address this problem by observing that if the fundamental solutions in Step 2 (denoted as φ_i^{in}) and Step 4 (denoted as φ_j^{out}) are predefined, then we can precompute the results of Step 2-Step 5 for an object. Similar to the BRDF for light, one can define the BRDF for sound. The mapping of φ_i^{in} to φ_j^{out} for an object is called the *per-object transfer function*. For different R^{in} that define an incident field p_{inc} on the interface, we only need to compute the expansion coefficients d_i of the fundamental solutions φ_i^{in}; the outgoing rays are computed by applying the precomputed per-object transfer function.

The outgoing scattered field that is modeled as outgoing rays from an object A may, after propagating in space and interacting with the environment, enter as incoming rays into the near-object region of another object B. For a scene where the environment and relative positions of various objects are fixed, we can precompute all the propagation paths for rays that correspond to A's outgoing basis functions $\varphi_{j,A}^{\text{out}}$ and that reach B's near-object region. These rays determine the incident pressure field arriving at object B, which can again be expressed as a linear combination of a set of basis functions $\varphi_{i,B}^{\text{in}}$. The mapping from $\varphi_{j,A}^{\text{out}}$ to $\varphi_{i,B}^{\text{in}}$, called the *inter-object transfer function*, which is a fixed function and can also be precomputed. Interactions between multiple objects can therefore be found by a series of applications of the inter-object transfer functions.

Based on the per-object and inter-object transfer functions, all orders of acoustic interaction (corresponding to multiple entrance of rays to near-object regions) in the scene can be found for the total sound field by solving a global linear system, which is much faster than the straightforward hybridization, where the underlying numerical solver is invoked multiple times for each order of interactions. The trade-off is that the transfer functions have to be precomputed. However, the pre-object transfer functions can be reused even when the objects are moved. This characteristic is beneficial for quick iterations when authoring scenes, and can potentially be a cornerstone for developing sound propagation systems that supports fully dynamic scenes.

Scene	#src	#freq.	#eq. srcs	Hybrid Pressure Solving								Pressure Evaluation
				Numerical				Geometric				
				wave sim.	per-object	inter-object	source + global	#tris	order	#rays	prop. time	
Building+small	1	300	220 K	163 min	552 min	22 min	19 min	60	3	4096	41 min	81 sec
Building+medium	1	400	290 K	217 min	736 min	33 min	23 min	60	3	4096	39 min	81 sec
Building+large	1	800	580 K	435 min	1472 min	54 min	40 min	60	3	4096	39 min	81 sec
Reservoir	1	500	500 K	254 min	252 min	4 min	2.6 min	16505	2	262144	1.9 min	10 sec
Parking	2	500	123 K	55 min	40 min	3 min	0.9 min	5786	2	4096	6.6 min	24 sec

Table 5.1: **Precomputation Performance Statistics.** The rows "Building+small", "Building+medium", and "Building+large" correspond to scenes with a building surrounded by small, medium, and large walls, respectively. "Reservoir" and "Parking" denote the reservoir and underground parking garage scene respectively. For a scene, "#src" denotes the number of sound sources in the scene, "#freq." is the number of frequency samples, and "#eq. srcs" denotes the number of equivalent sources. The first part, "Hybrid Pressure Solving". includes all the steps required to compute the final equivalent source strengths, and is performed once for a given sound source and scene geometries. The second part, "Pressure Evaluation", corresponds to the cost of evaluating the contributions from all equivalent sources at a listener position and is performed once for each listener position. For the numerical technique, "wave sim." refers the total simulation time of the numerical wave solver for all frequencies; "per-object" denotes the computation time of for per-object transfer functions; "inter-object" is the inter-object transfer functions for each pair of objects (including self-inter-object transfer functions, where the pressure wave leaves a near-object region and reflected back to the same object); "source + global " is the time to solve the linear system to determine the strengths of incoming and outgoing equivalent sources. For the geometric technique, "# tris" is the number of triangles in the scene; "order" denotes the order of reflections modeled; "# rays" is the number of rays emitted from a source (sound source or equivalent source). The column "prop. time" includes the time of finding valid propagation paths and computing pressures for any intermediate step (e.g. from one object to another object's offset surface).

5.3 Implementation

In this section we discuss the implementation aspect for our technique.

5.3.1 Implementation details

The geometric acoustics code is written in C++, based on the Impulsonic Acoustect SDK[2], which implements a ray-tracing based image source method. For the numerical acoustic technique we use a GPU-based implementation of the ARD wave-solver (Raghuvanshi et al., 2009b). Per-object transfer functions, inter-object transfer functions, and equivalent source strengths are computed using a MATLAB implementation based on (Mehra et al., 2013).

Table 5.1 provides the detailed timing results for the precomputation stage. The timings are divided into two groups. The first group, labeled as "Hybrid Pressure Solving," consists of all the steps required to compute the final equivalent source strengths. These computations are performed once for a given scene. The second group, labeled as "Pressure Evaluation," involves the computation of the pressures contributed by all equivalent sources at a listener position. This computation is performed once for each sampled listener position.

The timing results for "wave sim." (simulation time of the ARD wave solver), and "Pressure Evaluation" are measured on a single core of a 4-core 2.80 GHz Xeon X5560 desktop with 4GB of RAM and NVIDIA GeForce GTX 480 GPU with 1.5 GB of RAM. All the other results are measured on a cluster containing a total of 436 cores, with sixteen 16-CPUs (8 dual-core 2.8GHz Opterons, 32GB RAM each) and forty-five 4-CPU (2 dual-core 2.6GHz Opterons, 8GB RAM each).

We assume the scene is given as a collection of objects and terrains. In the spatial decomposition step, the offset surface is computed using distance fields. One important parameter is the spatial Nyquist distance h, corresponding to the highest frequency simulated $\nu_{\max}$, $h = c/2\nu_{\max}$, where c is the speed of sound. To ensure enough spatial sampling on the offset surface, we choose the voxel resolution of distance field to be h, and the sample points are the vertices of the surface given by the marching cubes algorithm. The offset distance is chosen to be $8h$. In general, a larger offset distance means a larger spatial domain for the numerical solver and is therefore more expensive. On the other hand, a larger offset distance results in a pressure field with less detail (i.e. reduced spatial variation)

[2] `http://impulsonic.com/acoustect-sdk/`

on the offset surface, and fewer outgoing equivalent sources are required to achieve the same error threshold.

5.3.2 Collocated equivalent sources

The positions of outgoing equivalent sources can be generated by a greedy algorithm that selects the best candidate positions randomly (James et al., 2006b). However, if each frequency is considered independently, a total of $1M$ or outgoing equivalent sources may arise across all simulated frequencies. Because we must trace N_r rays, (typically thousands or more) from each equivalent source, this computation becomes a major bottleneck in our hybrid framework. This may cause a computation bottleneck in our hybrid framework, because we need to trace N_r rays (typically thousands or more) from each equivalent source.

We resolve this issue by reusing equivalent sources positions across different frequencies as much as possible. First, the equivalent sources for the highest frequency $\nu_{\max}$, which requires the highest number of equivalent sources, $P_{\max}$, are computed using the greedy algorithm. For lower frequencies, the candidate positions are drawn from the $P_{\max}$ existing positions, which guarantees that a total of $P_{\max}$ collocated positions is occupied. Indeed, when the path is frequency-independent, rays emitted from collocated sources will travel the same path, which reduces the overall ray-tracing cost. The frequency-independent path assumption holds for paths containing only specular reflections, in which case the incident and reflected directions are determined. We observe a $60 - 100X$ speedup while maintaining the same error bounds over methods without the collocation scheme. All the timings results in this section are based on this optimization.

5.3.3 Auralization

We compute the frequency responses using our spatial decomposition approach up to $\nu_{\max} =$ 1 kHz with a sampling step size of 2.04 Hz. For frequencies higher than $\nu_{\max}$, we use a ray tracing solution, with diffractions approximated by the Uniform Theory of Diffraction (UTD) model (Kouyoumjian and Pathak, 1974). We join the low- and high-frequency responses in the region [800, 1000] Hz using a low-pass–high-pass filter combination.

The sound sources in our system are recorded audio clips. The auralization is performed using overlap-add STFT convolutions. A ”dry” input audio clip is first segmented into overlapping frames,

Scene	#IR samples	Memory	Time
Building+small	960	19 MB	3.5 ms
Building+med	1600	32 MB	3.5 ms
Building+large	6400	128 MB	3.5 ms
Reservoir	17600	352 MB	1.8 ms

Table 5.2: **Runtime Performance on a Single Core.** For each scene, "#IR samples" denotes the number of IR's sampled in the scene to support moving listeners or sources; "Memory" shows the memory to store the IR's; "Time" is the total running time needed to process and render each audio buffer.

and a windowed (Blackman window) Short-Time Fourier transform (STFT) is performed. The transformed frames are multiplied by the frequency responses corresponding to the current listener position. The resulting frequency-domain frames are then transformed back to time-domain frames using inverse FFT, and the final audio is obtained by overlap-adding the frames. For spatialization we use a simplified spherical head model with one listener position for each ear. Richer spatialization can be modeled using *head related transfer functions* (HRTFs), which are easily integrated in our approach.

For the interactive auralization we implemented a simplified version of the system proposed by Raghuvanshi et al. (2010). Only the listener positions are sampled on a grid (of 0.5m-1m grid size), and the sound sources are kept static. The case of moving sound sources and a static listener is handled using the principle of acoustic reciprocity (Pierce, 1989). The interactive auralization is demonstrated through integration with Valve's Source™game engine. Audio processing is performed using FMOD at a sampling rate of 44.1 kHz; the audio buffer length is 4096 samples, and the FFTs are computed using the Intel MKL library. The runtime performance statistics are summarized in Table 5.2. The parking garage scene is rendered off-line and not included in this table.

5.4 Results and Analysis

In this section we present the results of our hybrid technique in different scenarios and error analysis.

5.4.1 Scenarios

We demonstrate the effectiveness of our technique in a variety of scenes as shown in Figure 5.4. These scenes are at least as complex as those shown in previous wave-based sound simulation techniques (James et al., 2006b; Raghuvanshi et al., 2009b; Mehra et al., 2013) or geometric methods with precomputed high-order reverberation (Tsingos, 2009; Antani et al., 2012). Please refer to the supplementary video for the auralizations. In each scene, we compare the audio generated by our method with existing sound propagation methods: a pure geometric technique is used for comparison (Taylor et al., 2012), which models specular reflection as well as edge diffraction through UTD; a pure numerical technique, the ARD wave-solver (Raghuvanshi et al., 2009b). Comparisons with ARD are done only in a limited selection of scenes (Building), while the other scenes (Underground Parking Garage and Reservoir) are too large to fit in the memory using ARD.

Building. As the listener walks behind the building, we observe the low-pass occlusion effect with smooth transition as a result of diffraction. We also observe the reflection effects due to the surrounding walls. We show how sound changes as the distance from the listener to the walls and the height of the walls vary.

Underground Parking Garage. This is a large indoor scene with two sound sources, a human and a car, as well as vehicles that scatter and diffract sound. As the listener walks through the scene, we observe the characteristic reverberation of a parking garage, as well as the variation of sound received from various sources depending on whether the listener is in the line-of-sight of the sources.

Reservoir. We demonstrate our system in a large outdoor scene from the game Half-Life 2, with a helicopter as the sound source. This scene shows diffraction and scattering due to a rock; it also shows high-order interactions between the scattered pressure and the surrounding terrain, which is most pronounced when the user walks through a passage between the rock and the terrain. Interactive auralization is achieved by precomputing the IRs at a grid of predefined listener positions. We also make the helicopter fly and thereby demonstrate the ability to handle moving sound sources and high-order diffractions.

Figure 5.4: Our hybrid technique is able to model high-fidelity acoustic effects for large, complex indoor or outdoor scenes at interactive rates: (a) building surrounded by walls, (b) underground parking garage, and (c) reservoir scene in Half-Life 2.

5.4.2 Error Analysis

In Figure 5.5 we compare the results of our hybrid technique with BEM on a spatial grid of listener locations at different frequencies for several scenes: two parallel walls, two walls with a ground, an empty room, and two walls in a room. BEM is one of the most accurate wave-based simulators available, and comparing with high-accuracy simulated data is a widely adopted practice (Barbone et al., 1998; Jean et al., 2008; Hampel et al., 2008). BEM results are generated by the FastBEM simulator[3]. A comparison with a geometric technique for the last scene is also provided. The geometric technique models 8 orders of reflection and 2 orders of diffraction through UTD.

We also compute the difference in pressure field (i.e. the error) between our hybrid technique with varying reflection orders and BEM, as shown in Figure 5.6 for the "Two Walls in a Room" scene. The error between the pressure fields generated by the reference wave solver and by our hybrid method , is computed as $||P_{\text{ref}} - P_{\text{hybrid}}||^2/||P_{\text{ref}}||$, where P_{ref} and P_{hybrid} are vectors consisting of complex pressure values at all the listener positions and $||\cdot||$ denotes the two-norm of complex values, summed over all positions $\mathbf{x}$ (the grid of listeners as shown in Figure 5.5). Higher reflection orders lead to more accurate results but require more rays to be traced.

5.4.3 Complexity

Consider a scene with κ objects. We perform the complexity analysis for frequency ν and discuss the cost of numerical and geometric techniques used.

Numerical Simulation and Pre-Processing: The pre-processing involves several steps: (1) performing the wave simulation using numerical techniques, (2) computing per-object and inter-object

[3] `http://www.fastbem.com/`

transform matrix, and (3) solving linear systems to determine the strengths of incoming and outgoing equivalent sources (Mehra et al., 2013). In our system, the equivalent sources are limited to monopoles and dipoles, and the complexity follows:

$$O(\kappa nQP^2 + \kappa^2 nPQ^2 + \kappa(u \log u) + \kappa^3 P^3), \tag{5.16}$$

where Q, P are the number of incoming and outgoing equivalent sources respectively, n is the number of offset surface samples, and u is the volume of an object. The number of equivalent sources P and Q scale quadratically with frequency.

Ray Tracing: Assume the scene has T triangles, and from each source we trace N_r rays to the scene. The cost for one bounce of tracing from a source is $O(N_r \log T)$ on average and $O(N_r T)$ in the worst case. If the order of reflections modeled is d, then the (worst case) cost of ray-tracing is $O(N_r T^d)$. This cost is multiplied by the number of sources (sound sources and equivalent sources) and the number of points where the pressure values need to be evaluated. The total cost is dominated by computing inter-object transfer functions, where the pressure from P outgoing equivalent sources from an object needs to be evaluated at n sample positions on the offset surface of another object. This results in

$$O(\kappa^2 PnT^d) \tag{5.17}$$

for a total of κ^2 pairs of objects in the scene.

In our collocated equivalent source scheme, however, the P outgoing sources for different frequencies share a total of P_{col} positions. The rays traced from a shared position can be reused, so for all frequencies ν, we only need to trace rays from P_{col} positions instead of $\sum_\nu P(\nu)$ positions .

The choice of N_r is scene-dependent. In theory, in order to discover all possible reflections from all scene triangles without missing a propagation path, the ray density along every direction should be high enough so that the triangle spanning the smallest solid angle viewed from the source can be hit by at least one ray. The problem of missing propagation paths is intrinsic to all ray-tracing methods. It can be overcome by using beam-tracing methods (Funkhouser et al., 1998), but they are considerably more expensive and are only practical for simple scenes.

The order of reflection d also depends on the scene configuration. For an outdoor scene where most reflections come from the ground, a few reflections are sufficient. In enclosed or semi-enclosed spaces more reflections are needed. In practice it is common to stop tracing rays when a given bound of reflection is reached, or when the reflected energy is less than a threshold.

Scalability Although the computation domain of the numerical solver, Ω^N, is smaller than the entire scene, the size of the entire scene still matters. Larger scenes require longer IR responses and therefore more frequency samples, which affect the cost of both numerical and geometric components of our hybrid approach. Larger scenes in general require more triangles, assuming the terrain has the same *feature density*. For a scene whose longest dimension is L, the number of IR samples (and therefore frequency samples) scales as $O(L)$, and the number of triangles scales as $O(L^2)$, - giving overall numerical and ray-tracing complexities of - $O(L)$ and $O(L^3 \log L)$ respectively. This is better than most numerical methods; for example, the time complexity of ARD are $O(L^4 \log L)$ and FDTD scale $O(L^4)$.

We tested the scalability of our method with the building scene by increasing the size of the scene and measuring the performance. The results are shown in Figure 5.7. Since the open space is handled by geometric methods, whose complexity of the geometric method is not a direct function of the total volume, it is not necessary to divide the open space into several connected smaller domains, as some previous methods did (Raghuvanshi et al., 2009b).

5.4.4 Comparison with Prior Techniques

Compared with geometric techniques, our approach is able to capture wave effects such as scattering and high-order diffraction, thereby generate sound of higher quality. When compared with performing numerical wave-based techniques such as ARD and BEM, over the entire domain, our approach is much faster as we use a numerical solver only in near-object regions, as opposed to the entire volume. We do not have a parallel BEM implementation, but extrapolating from the data in Figure 6, FastBEM would take 100+ hours for Underground Parking Garage and 1000+ hours for Reservoir on a 500-core cluster to simulate sound up to 1 kHz, assuming full parallelization. In comparison, our method can perform all (numeric, geometric, and coupling) precomputations in a few hours for these two scenes (as shown in Table 5.1) to achieve interactive runtime performance (see Table 5.2). Moreover, numerical techniques typically require memory proportional to the third

or fourth power of frequency to evaluate pressures and compute I's at different listener positions. As shown in Table 5.3, our method requires orders of magnitude less memory than several standard numerical techniques. We have also highlighted the relative benefits of our two-way coupling algorithms with other hybrid methods used in acoustic and electromagnetic simulation (see Section 2.3). In many ways, our coupling algorithm ensures continuity and consistency of the field computed by numeric and geometric techniques at the artificial boundary between their computational domains.

The method proposed by Mehra et al. (2013) is also able to simulate the acoustic effects of objects in large outdoor scenes. Their formulation, however, only allows objects to be situated in an empty space or on an infinite flat ground, and therefore cannot model large indoor scenes (e.g. parking lot) or outdoor scenes with uneven terrains. If an outdoor scene has a large object, the algorithm proposed in (Mehra et al., 2013) would slow down considerably. The coupling with geometric propagation algorithm, on the other hand, enables us to model acoustic interactions with all kinds of environment features. It is relatively easier to extend our hybrid approach to inhomogeneous environments by using curved ray tracing. Furthermore, geometric ray tracing is also used to perform frequency decomposition and this results in improved sound rendering.

Scene	air vol. (m^3)	surf. area (m^2)	FDTD	ARD	BEM/ FMM	Ours
Bldg+small	1800	660	0.2 TB	5 GB	6 GB	**12 MB**
Bldg+med	3200	1040	0.3 TB	9 GB	9 GB	**12 MB**
Bldg+large	22400	3840	2.2 TB	60 GB	34 GB	**12 MB**
Reservoir	5832000	32400	578 TB	16 TB	307 GB	**42 MB**
Parking	9000	2010	0.9 TB	24 GB	2 GB	**9 MB**

Table 5.3: **Memory Cost Saving.** The memory required to evaluate pressures at a given point of space. This corresponds to the same operation shown in the rightmost column of Table 5.1. Compared to standard numerical techniques, our method provides **3 to 7 orders of magnitude** of memory saving on the benchmark scenes.

5.5 Limitations, Conclusion, and Future Work

We have presented a novel hybrid technique for sound propagation in large indoor and outdoor scenes. The hybrid technique combines the strengths of numerical and geometric acoustic techniques for the different parts of the domain: the more accurate and costly numerical technique is used to model wave phenomena in near-object regions, while the more efficient geometric technique is used

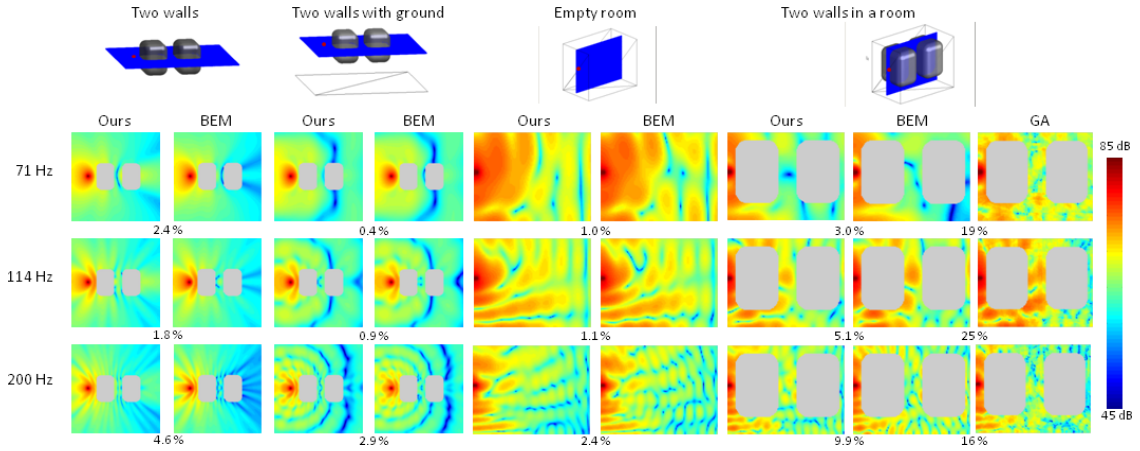


Figure 5.5: Comparison between the magnitude of the total pressure field computed by our hybrid technique and BEM for various scenes. In the top row, the red dot is the sound source, and the blue plane is a grid of listeners. Errors between our method and BEM for each frequency are shown in each row. For our hybrid technique, the effect of the two walls are simulated by numerical acoustic techniques, and the interaction between the ground or the room is handled by geometric acoustic techniques. For BEM, the entire scene (including the walls, ground, and room) is simulated together. The last column also shows comparison with a pure geometric technique (marked as "GA").

to handle propagation in far-field regions and interaction with the environment. The sound pressure field generated by the two techniques is coupled using a novel two-way coupling procedure. The method is successfully applied to different scenarios to generate realistic acoustic effects.

Our approach has a few limitations. The diffraction due to objects is currently handled completely by the numerical component in the near-object regions of our hybrid system. It is possible to also include geometric approximations of the diffraction effect, such as the UTD or BTM methods, in the far-field regions. This approach offers flexibility to determine how accurately the diffraction effects should be modeled, where and when numerical methods should be approximated by geometric methods.

The performance of our spatial decomposition depends greatly on the size of Ω^N. Although it size is smaller than the entire simulation domain, an individual Ω^N may still be too large, especially when the wave effects near a large object need to be computed and this increases the complexity of our algorithm. One interesting topic to investigate is the possibility of not enclosing the whole object, but only parts of it (e.g. small features) in Ω^N.

We currently compare our simulation results with simulated data from a high-accuracy BEM solver. It would be an important future work to validate these results with recorded audio measure-

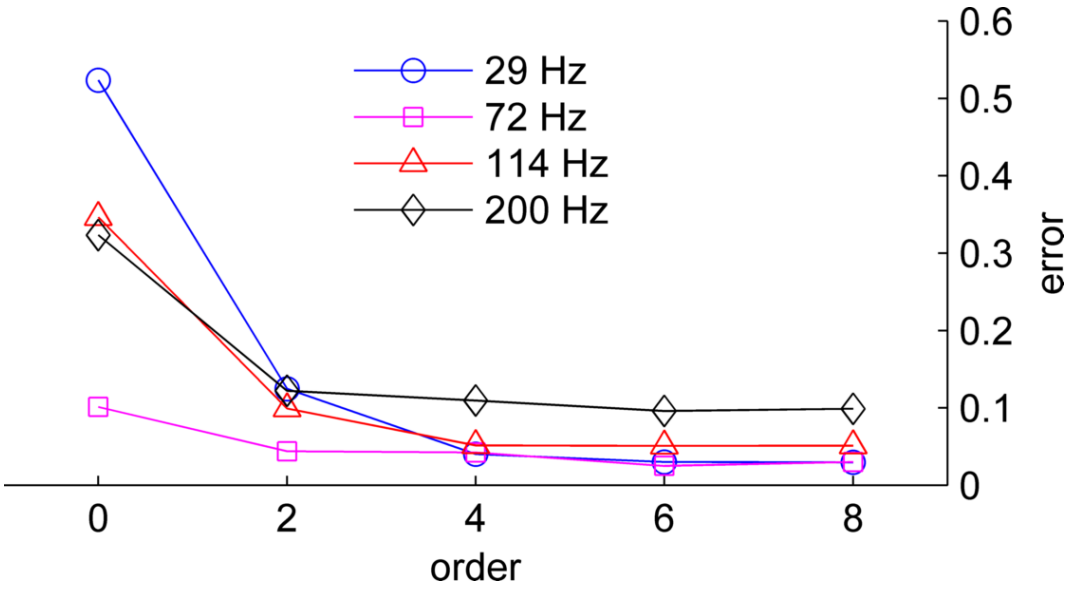


Figure 5.6: Error $\|P_{\text{ref}} - P_{\text{hybrid}}\|^2 / \|P_{\text{ref}}\|$ between the reference wave solver (BEM) and our hybrid technique for varying maximum order of reflections modeled. The tested scene is the "Two walls in a room" (see also Figure 5.5, last column).

ments, when accurate measurements with binaural sound recordings and spatial sampling in complex environments are available.

Additionally our approach and system implementation is currently limited to mostly static scenes with moving sound sources and/or listeners. Nonetheless the use of transfer functions lays the foundation for future extension to fully dynamic scenes, as the per-object transfer functions of an object can be reused even when the object is moved. In order to recompute inter-object transfers as multiple objects move in a dynamic scene, a large number of rays (the number of outgoing sources for all frequency samples multiplied by thousands of rays emitted per source) need to be retraced. We would like to explore the use of the Fast Multipole Method (FMM) (Gumerov and Duraiswami, 2004) to reduce the number of outgoing sources for far-field approximations. The computation of transfer function is currently implemented with unoptimized MATLAB code, and using high-performance linear solvers (CPU- or GPU-based) can greatly improve the performance.

5.6 Extension to Inhomogeneous Media

In previous sections, my geometric technique assumes homogeneous media and traces straight ray paths. In real world, however, the media in which sound travels is usually not homogeneous:

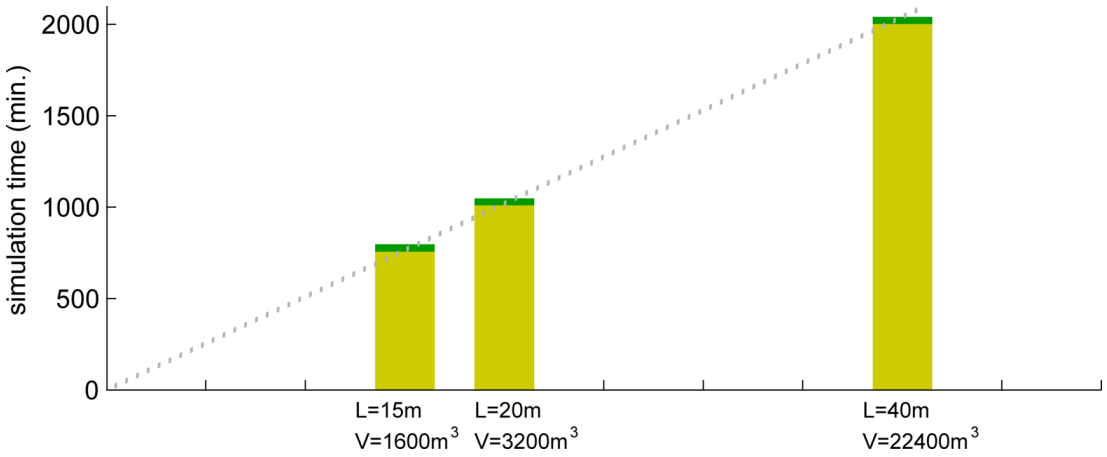


Figure 5.7: **Breakdown of Precomputation Time.** For a building placed in terrains of increasing volumes (small, medium, and large walls), the yellow part is the simulation time for the numerical method, and the green part is for the geometric method. The numerical simulation time scales linearly to the largest dimension (L) of the scene instead of the total volume (V).

there is wind, temperature difference, turbulence in the atmosphere, as well as salinity difference underwater – all cause the speed of sound to vary in space. The deviation from the homogeneous approximation becomes non-negligible for large scenes (e.g. spanning kilometers). In this section I discuss the extension to inhomogeneous medium, where the speed of sound is not constant and the rays may travel in curved paths. A curved ray-tracing module must be integrated into my hybrid system instead. The major challenge of extending from homogeneous to inhomogeneous medium is the presence of a kind of irregularities called *caustic points*. The standard Ray Theory fails to predict physically meaningful results around these irregularities and special treatments need to be taken. Even the first step–identifying their locations in space is challenging. Previously several methods that aim to locate these points and introduce correction terms to the standard Ray Theory are proposed (Ludwig, 1966; Salomons, 2001), but even if they only solve the reduced two dimensional problem (which is useful if the media variation is azimuth-symmetrical) the methods are quite intricate. The problem only worsen in the case of full three-dimensional problem, which is actually needed in many real-world sound propagation applications (Tolstoy, 1996).

The rest of this section is therefore mostly devoted to overcome such challenges and are organized as follows. First, in order to understand the difficulties and necessary theoretical modifications when extending to inhomogeneous medium, the standard Ray Theory is revisited in Section 5.6.1. I show that the Ray Theory originates from solving the acoustic wave equation,which can be decomposed to

two equations under high-frequency approximation: the eikonal equationand the transport equation.I will show that the eikonal equation determines the ray trajectory, which has analytical solutions in some special cases. The transport equation, on the other hand, is related to the pressure amplitude on a ray. By introducing several coordinate transforms, I examine some geometrical properties of rays (e.g. the cross-sectional area of a ray tube) and establish the relationship between these properties and the amplitude. Under this mathematical framework, it is then clear what a caustic point is, where it would occur, and why it causes the standard Ray Theory to fail. The failure can be discussed in two aspects: one is related to the infinite (and therefore unphysical) amplitude that the standard Ray Theory predicts at caustic points; the other is related to the phase inversion across caustic points. The first problem is treated in Section 5.6.3 and 5.6.4, and the second is solved in Section 5.6.2.

With the theoretical background of Section 5.6.1, I then discuss the computational aspect in detail in Section 5.6.2, namely how to solve the eikonal equation and the transport equation by tracking extra variables (mostly related to the geometrical properties of rays) along ray paths. The computation of coordinate transforms that are necessary for obtaining these geometrical properties are explained step by step.

After Section 5.6.2, the pressure field at any point (with the exception of a caustic point) along a ray can be computed. In theory if I wish to evaluate the pressure field at any point in space, I must find the *exact* ray passing through this point. The search for such rays, however, is very challenging in a three-dimensional space. Therefore I adopt a mathematical tool called *Gaussian Beams* which is developed in the seismology field (Popov, 1982; Červeny̌ et al., 1982) and then extended to the acoustics field (Porter and Bucker, 1987). The Gaussian Beam method essentially associates a non-zero width to each ray, and thereby extends the pressure field to points not on a ray. It also eliminates the problem of infinite amplitude at caustic points. The pressure field at any given point can thus be computed by first finding the nearby rays passing through the vicinity of the point (avoiding the search of the ray passing *exactly* through that point), and then computing the weighted sum of their contributions, The weighting function, as well as the computation of other necessary components, are carefully investigated in Section 5.6.4.1. Combining all these components, the final pressure field can be computed using Equation (5.77).

I adopted most of the mathematical results from works by Červený (Červený, 2000, 2005). Detailed derivations are omitted here, and interested readers are referred to his works. His theory is

intentionally presented in a very general form so that it can be applied many kinds of mechanical waves, including acoustic and seismic waves. In my discussion I present specialized forms tailored to acoustic applications and also elaborate the computational considerations that comes with these applications.

Due to the complicated nature of the problem at hand, and the necessity to introduce several coordinate transforms as discussed previously, there are many mathematical symbols in this section. A list of symbols and their meanings is provided in Table 5.4. Please note that cases and styles all matter, so P, p, $\vec{p}$, and $\mathbf{P}$ all have different meanings. In order to improve readability, however, I follow a set of strict, consistent conventions for the mathematical notations as suggested by Červený (Červený, 2005). Matrices are all bold-faced ($\mathbf{M}$), and vectors are denoted with arrows ($\vec{v}$). To distinguish between 2×2 matrices and matrices of other dimensions, the circumflex ($\hat{}$) are used for 3×3 and 3×2 matrices. Components of matrices or vectors are always indexed in the form of suffixes. The uppercase suffixes take the values 1 and 2, lowercase indices 1, 2, and 3. In this way, M_{IJ} denote elements of $\mathbf{M}$ and M_{ij} elements of $\hat{\mathbf{M}}$. Sometimes when referring to components, I use a shorthand of x_i instead of writing all 3 components out, so that $f(x_i)$ actually means $f(x_1, x_2, x_3)$. The Einstein summation convention is used throughout this part of my thesis, where repeated indices imply that a summation is taken. Thus $M_{IJ}q_J = M_{I1}q_1 + M_{I2}q_2$ ($I = 1$ or 2), $M_{ij}q_j = M_{i1}q_1 + M_{i2}q_2 + M_{i3}q_3$ ($i = 1, 2$ or 3).

5.6.1 Ray Theory

In order to modify the ray-tracing module to incorporate inhomogeneous media, I shall revisit the theoretical background of ray tracing as a sound propagation method, what problem it tries to solve and how it should be modified.

Ray-tracing aims to solve the acoustic wave equation. Let us consider an acoustic wave equation for pressure p without source term,

$$\nabla \cdot \frac{1}{\rho} \nabla p = \frac{1}{\rho c^2} \ddot{p}. \tag{5.18}$$

For inhomogeneous media, both the sound velocity c and density ρ are variable. I can find an approximate time-harmonic (i.e. frequency-dependent) high-frequency solution of this equation in

symbol	meaning
p	pressure
x_i, (x_1, x_2, x_3)	components of Cartesian coordinates
ρ	density of the medium
c	speed of sound
P	pressure amplitude
ω	angular frequency of a sound wave
T	travel time function
$\vec{p}$	slowness vector
p_i	components of a slowness vector
$V(x_i)$	speed of sound written explicitly in a space-varying form
$\mathcal{H}$	Halmitonian
u	an arbitrary monotonic parameter along a ray
s	arclength along a ray
σ	a monotonic parameter chosen so that $\mathrm{d}\sigma = V\,\mathrm{d}s$
$\vec{A}$	gradient of V^{-2}
γ_1, γ_2	abstract parameters describing a ray; for example the initial take-off angles
i_0, ϕ_0	initial take-off angles of a ray
q_1, q_2, q_3	components of the ray-centered coordinates
$\vec{e}_1$, $\vec{e}_2$, $\vec{e}_3$	unit basis vectors of the ray-centered coordinates
$\hat{\mathbf{H}}$	a 3×3 transformation matrix from ray-centered coordinates to Cartesian coordinates
H_{ik}	matrix elements of matrix $\hat{\mathbf{H}}$
$p_i^{(q)}$	components of slowness vector in ray-centered coordinates
$\mathbf{Q}$, $\mathbf{P}$	2×2 matrices; see Equation (5.31) for definition
J	ray Jacobian
$\mathcal{L}$	geometrical spreading; defined as $\lvert J\rvert^{1/2}$
$\vec{t}$	unit vector tangent to the ray
$k(R, S)$	KMAH index from point S to point R
$\mathbf{M}$	2×2 matrix; the second derivative of the travel-time field with respect to q_1 and q_2
$\hat{\mathbf{Q}}^{(x)}$, $\hat{\mathbf{P}}^{(x)}$	3×2 transform matrices; see Equation (5.41) for definition
$\Sigma_{/\!/}$	plane where a ray lies
$\vec{n}_1$, $\vec{n}_2$, $\vec{n}_3$	a set of orthonormal unit vectors defined in relationship with a ray and the plane $\Sigma_{/\!/}$ that it lies in
$T^c(R, S)$	phase shift due to caustics between point S and R
P^{ray}	ray amplitude
$\Phi(\gamma_1, \gamma_2)$	weighting function of the contribution of a ray described by parameters γ_1, γ_2
$\mathcal{D}$	domain of ray parameters under consideration
$\boldsymbol{\mathcal{M}}$	2×2 matrix defined by Equation (5.68)
$\hat{\mathbf{M}}^{(x)}$	3×3 matrix related to $\mathbf{M}$; see Equation (5.71)

Table 5.4: **Symbol Table.**

the following form (Jensen et al., 2011):

$$p(x_i, \omega, t) = P(x_i) \exp[-\mathrm{i}\omega(t - T(x_i))]. \tag{5.19}$$

ω is the angular frequency of the sound wave. x_i is a short-hand for (x_1, x_2, x_3) and denotes a point in space. $T(x_i)$ is a smooth scalar functions of coordinates, representing the time for the wave to travel from source to point x_i in space, and is often referred to as the *travel time function*. $P(x_i)$ is a time-independent *pressure amplitude function*, which is also space-varying. Notice Equation (5.19) is just performing separation of variables for the pressure function $p(x_i, \omega, t)$, I have not introduce any physics yet.

Substituting this equation to Equation (5.18), I obtain:

$$\begin{aligned} &- \omega^2 \left[(\nabla T)^2 - \frac{1}{c^2} \right] \\ &+ \mathrm{i}\omega \left[2\nabla P \cdot \nabla + P\nabla^2 T - \left(\frac{P}{\rho} \right) \nabla T \cdot \nabla \rho \right] \\ &+ \rho \nabla \cdot \frac{1}{\rho} \nabla P = 0. \end{aligned} \tag{5.20}$$

Because Equation (5.20) must be satisfied for any frequency ω, the expressions with ω^2, ω^1, and ω^0 must vanish. For high frequencies, $\omega \gg 0$. the most important terms will be the term with ω^2 and ω^1, corresponding to the first and second terms in Equation (5.20). These two terms should vanish, thus giving us the *eikonoal equation*,

$$(\nabla T)^2 = 1/c^2, \tag{5.21}$$

and the *transport equation*,

$$2\nabla P \cdot \nabla T + P\nabla^2 T - (P/\rho)\nabla T \cdot \nabla \rho = 0. \tag{5.22}$$

These two equations are fundamental in the ray theory for solving the acoustic wave equation. The eikonal equation is a nonlinear partial differential equation of the first order for travel time $T(x_i)$. It is usually solved by *ray tracing*. The transport equation is a linear partial differential equation of the

first order in $P(x_i)$ and can be solved quite simply along the rays. In the following two subsections I shall discuss how to solve these two equations respectively.

5.6.1.1 Solving the Eikonal Equation

The eikonal equation $(\partial T)^2 = 1/c^2$ is a nonlinear partial differential equation of the first order for travel time $T(x_i)$. I introduce a *slowness vector* $\vec{p} = \nabla T$ (not to be confused with pressure p), which is the spatial derivative of the travel time field T. The name *slowness* is from the seismology literature (Červený, 2005) and comes from the fact that its magnitude is the inverse of the speed of sound, $|\vec{p}| = 1/c$. In Cartesian coordinates the components are $p_i = \partial T/\partial x_i$, and the eikonal equation reads

$$p_i p_i = 1/V^2(x_i). \tag{5.23}$$

Here $V(x_i) = c$ is the space-varying sound speed. Equation (5.23) can be written in the *Hamiltonian* form:

$$\mathcal{H}(x_i, p_i) = p_i p_i - 1/V^2(x_i) = 0. \tag{5.24}$$

The name *Hamiltonian* comes from classical mechanics, where it represents the canonical equations of motion of a particle moving in the field governed by the Hamiltonian function $\mathcal{H}(x_i, p_i)$ and has energy $\mathcal{H} = 0$ (Goldstein, 1980).

In mathematics, the nonlinear partial differential equation is usually solved in terms of *characteristics*. The characteristics of Equation (5.24) are 3-D space trajectories $x_i = x_i(u)$ for u some parameter along the trajectory, along which $\mathcal{H}(x_i, p_i) = 0$ is satisfied. The detailed derivation of the characteristic system shall be neglected here, the reader is referred to textbooks (Bleistein, 1984). The characteristic system of the nonlinear partial differential equation (5.24) reads

$$\frac{\mathrm{d}x_i}{\mathrm{d}u} = \frac{\partial \mathcal{H}}{\partial p_i}, \qquad \frac{\mathrm{d}p_i}{\mathrm{d}u} = -\frac{\partial \mathcal{H}}{\partial x_i}, \qquad \frac{\mathrm{d}T}{\mathrm{d}u} = p_k \frac{\partial \mathcal{H}}{\partial p_k}, \qquad i = 1, 2, 3. \tag{5.25}$$

The solution of $x_i = x_i(u)$ is the characteristic curve as a 3-D trajectory, which is defined as a *ray*. The solution $p_i = p_i(u)$ are components of the slowness vector along the ray, and the travel time $T = T(u)$ can be solved along the ray. The system of ordinary differential equations (5.25) are called

ray tracing system. It shall be easy to see that once $\mathcal{H}(x_i, p_i) = 0$ is satisfied at one reference point of the characteristic (ray), it is satisfied along the whole ray.

The choice of parameter u depends on the specific form of function $\mathcal{H}$, and may take the form of travel time T, arclenght s along the ray, or a monotonic parameter σ, where $\mathrm{d}\sigma = V\,\mathrm{d}s$. A useful case is that if we choose u to be σ in the formulation of $\mathcal{H}$ (Equation (5.24)), then the ray tracing system reads

$$\frac{\mathrm{d}x_i}{\mathrm{d}\sigma} = p_i, \qquad \frac{\mathrm{d}p_i}{\mathrm{d}\sigma} = \frac{1}{2}\frac{\partial}{\partial x_i}\left(\frac{1}{V^2}\right), \qquad \frac{\mathrm{d}T}{\mathrm{d}\sigma} = \frac{1}{V^2}. \tag{5.26}$$

In this specially chosen case, I shall make a remark that if the media has a constant gradient of the square of slowness, V^{-2}, the ray tracing system (Equation (5.26)) has an analytical solution. Assume that V^{-2} is described by $V^{-2}(x) = A_0 + \vec{A}\cdot\vec{x}$, or written in components

$$V^{-2}(x_i) = A_0 + A_1 x_1 + A_2 x_2 + A_3 x_3. \tag{5.27}$$

A_0 is a the reference value of V^{-2} at the origin $x_1 = x_2 = x_3 = 0$, and $\vec{A}$ is the gradient of the square of slowness. In acoustics literature this corresponds to a $n^2 - linear$ media profile (Jensen et al., 2011), where n is the refraction index and is proportional to V^{-1}.

Plugging Equation (5.27) into Equation (5.26), the readers can verify that the analytical solution is then

$$\begin{aligned}
x_i(\sigma) &= x_{i0} + p_{i0}(\sigma - \sigma_0) + \tfrac{1}{4}A_i(\sigma - \sigma_0)^2,\\
p_i(\sigma) &= p_{i0} + \tfrac{1}{2}A_i(\sigma - \sigma_0),\\
T(\sigma) &= T(\sigma_0) + V_0^{-2}(\sigma - \sigma_0) + \tfrac{1}{2}A_i p_{i0}(\sigma - \sigma_0)^2 + \tfrac{1}{12}A_i A_i(\sigma - \sigma_0)^3.
\end{aligned} \tag{5.28}$$

Here the parameter σ along the ray is related to travel time T and to arclength s by $\mathrm{d}\sigma = V^2\,\mathrm{d}T = V\,\mathrm{d}s$. Hence, the ray is a parabolic curve.

The analytical solutions of the special case inspire *cell methods* (Jensen et al., 2011; Červený, 2005). The philosophy of cell methods is to divide the domain into subdomains called *cells*, typically tetrahedrons. Within each cell the media is fitted by some simple form, like the constant gradient V^{-2} described above, for which an analytic solution of the ray trajectory is possible. The ray can thus be traced inside a cell, and when it reaches the boundaries it would enter another cell. The

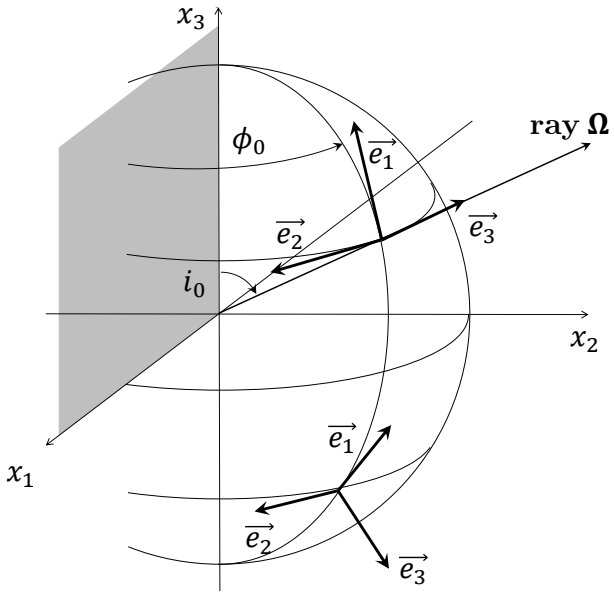


Figure 5.8: Initial take-off angles i_0 and ϕ_0 as ray parameters. i_0 is the angle between the ray direction and the x_3-axis, while ϕ_0 is the angle between the ray direction and the x_1-x_3 plane. $0 \leq i_0 \leq \pi$ and $0 \leq \phi_0 < 2\pi$. A possible choice of the initial basis vectors $\vec{e}_1$, $\vec{e}_2$, $\vec{e}_3$ of the ray-centered coordinate system are also plotted on the unit sphere.

whole trajectory of the ray can thus be analytically traced segment-by-segment within contiguous cells. In the tetrahedral cells, the velocity is continuous across the boundaries of the cells, therefore the ray trajectories are smooth (with C^1 continuity) across boundaries. I adopt this method, and the following discussion I assume cells are already fitted within which V^{-2} has a constant gradient.

5.6.1.2 Solving the Transport Equation

Before solving the transport equation (5.22), it is useful to discuss important concepts and properties of the ray field, such as ray parameters, the Jacobians, the ray tube, and geometrical spreading.

Consider an orthonormal system of rays from the same source, parameterized by two *ray parameters* γ_1, γ_2 (if the source is fixed, a ray's direction has two degrees of freedom). The parameters are used to discriminate each ray in a system of rays, and can be introduced in many ways. For example, for rays emitted from a point source, I may use the two take-off angles i_0 and ϕ_0

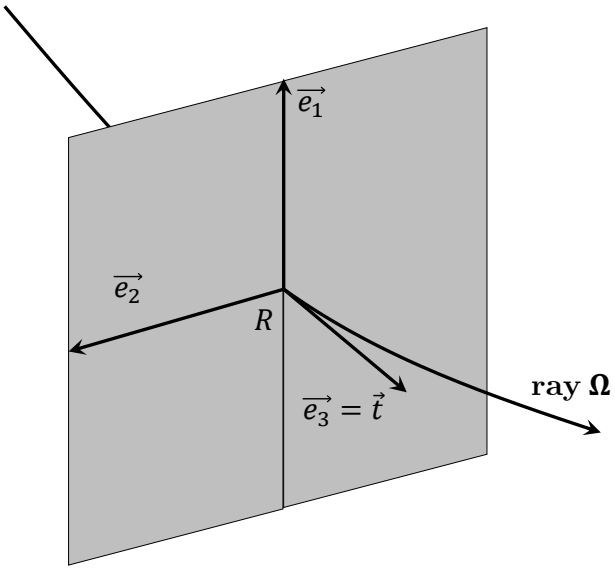


Figure 5.9: Basis vectors $\vec{e}_1$, $\vec{e}_2$, $\vec{e}_3$ of the ray-centered coordinate system q_i connected with ray Ω. Ray Ω is the q_3-axis of the system. At any point on the ray, unit vector $\vec{e}_3$ equals $\vec{t}$, the unit tangent to Ω. Unit vectors $\vec{e}_1$ and $\vec{e}_2$ are perpendicular to Ω and are mutually perpendicular.

as the ray parameters (see Figure 5.8). It would be possible to consider any other two parameters that specify the initial direction of the ray as the ray parameters.

At any point of ray Ω, I may also introduce the *ray-centered coordinates* q_1, q_2, q_3, with its origin at that point. Ray Ω is the q_3-axis of the system. I denote its unit basis vectors by $\vec{e_1}$, $\vec{e_2}$, $\vec{e_3}$. Unit vector $\vec{e_3}$ equals $\vec{t}$, the unit tangent to Ω. Unit vectors $\vec{e_1}$ and $\vec{e_2}$ are situated in a plane (shown as the shaded plane in Figure 5.9) , perpendicular to Ω at a given q_3, and are mutually perpendicular.

The 3×3 transformation matrix from the ray-centered cooridnates q_k to the Cartesian coordinates x_i is denoted by $\hat{\mathbf{H}}$, whose elements are

$$H_{ik} = \partial x_i/\partial q_k = \partial q_k/\partial x_i = e_{ki}, \tag{5.29}$$

where e_{ki} is the i-th Cartesian component of the unit vector $\vec{e}_k$. The 3×3 matrix $\hat{\mathbf{H}}$ can be used to express the slowness vector $\vec{p}$ in ray-centered components, denoted as $p_i^{(q)}$,

$$p_i^{(q)} = H_{ki} p_k. \tag{5.30}$$

The superscript (q) is used to hint that it is expressed in the ray-centered coordinates q_1, q_2, q_3. Note that since vector $\vec{p}$ is tangent to the ray and thus parallel to $\vec{e_3}$, I have $p_1^{(q)} = p_2^{(q)} = 0$.

Having defined the ray parameters and ray-centered coordinates, I am able to introduce the 2×2 matrices $\mathbf{Q}$ and $\mathbf{P}$, with elements

$$Q_{IJ} = (\partial q_I / \partial \gamma_J)_{T=\text{const.}}, \qquad P_{IJ} = (\partial p_I^{(q)} / \partial \gamma_J)_{T=\text{const.}}. \tag{5.31}$$

These matrices are very useful, and can be computationally determined along ray Ω once they are known at one point on Ω. The actual computation of these matrices will be discussed in detail in Section 5.6.2.

The determinant of $\mathbf{Q}$ is often denoted as J,

$$J = \det \mathbf{Q} \tag{5.32}$$

which is called *ray Jacobian*. It is the Jacobian of transformation from ray parameters γ_1, γ_2 to ray-centered coordinates q_1, q_2.

Jacobian J is closely connected with certain geometrical properties of the system of rays, particularly with the density of rays. Consider a *ray tube*, which is a family of rays, whose parameters are within the limits $(\gamma_1; \gamma_1 + \mathrm{d}\gamma_1)$ and $(\gamma_2; \gamma_2 + \mathrm{d}\gamma_2)$. See Figure 5.10. The cross-sectional area of $ABCD$ is proportional to $|J|^{1/2}$. The amplitudes of sound pressures are inversely proportional to $|J|^{1/2}$, as amplitudes are high in regions where the density of rays is high (small J), and in regions where the density of rays is low (high J), the amplitudes are low. Function $|J|^{1/2}$ is often called the *geometrical spreading* in the literature, and I denote it by $\mathcal{L}$.

The transport equation Equation (5.22) can be solved along rays for pressure amplitude P in terms of the ray Jacobian J. Using $P/\sqrt{\rho}$ instead of P in Equation (5.22), and noting that along the ray $\nabla T = c^{-1}\vec{t}$, where c is the space-varying sound speed and $\vec{t}$is the unit vector tangent to the ray, thus $\vec{t} \cdot \nabla(P/\sqrt{\rho}) = \mathrm{d}(P/\sqrt{\rho})\,\mathrm{d}s$, the transport function read can be rewritten as

$$\frac{\mathrm{d}}{\mathrm{d}s}\left(\frac{P}{\sqrt{\rho}}\right) + \frac{c}{2}\frac{P}{\sqrt{\rho}}\nabla^2 T = 0. \tag{5.33}$$

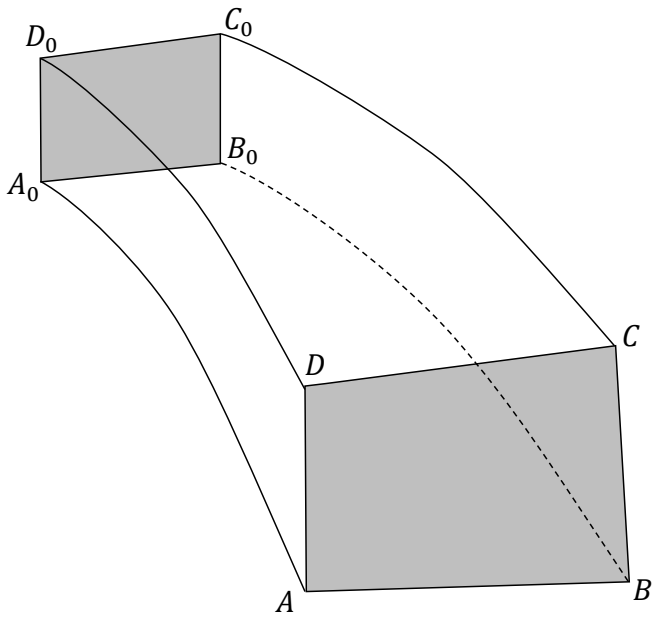


Figure 5.10: Ray tube. Ray A_0A corresponds to ray parameters (γ_1, γ_2), ray B_0B corresponds to $(\gamma_1 + \mathrm{d}\gamma_1, \gamma_2)$, ray C_0C corresponds to $(\gamma_1 + \mathrm{d}\gamma_1, \gamma_2 + \mathrm{d}\gamma_2)$, and ray D_0D corresponds to $(\gamma_1, \gamma_2 + \mathrm{d}\gamma_2)$.

The detailed derivation can be found in Červený (Červený, 2005). The solution of this equation is

$$P(s) = \left[\frac{\rho(s)c(s)J(s_0)}{\rho(s_0)c(s_0)J(s)}\right]^{1/2} P(s_0). \tag{5.34}$$

The amplitude $P(s)$ can be determined along the ray using Equation (5.34), once $P(s_0)$ is known at some reference point $s = s_0$ of the ray.

Equation (5.34) also gives us an insight of where *caustic points* appear. Caustic points, or simply *caustics*, are points of the ray, at which the ray Jacobian vanishes ($J = 0$), and the cross-sectional area of the ray tube shrinks to zero.

Since $J = \det \mathbf{Q}$, I can specify the position of caustic points along the ray by $\det \mathbf{Q} = 0$, which happens when the rank of the 2×2 matrix $\mathbf{Q}$ is less than 2. There are two types of caustic points along the ray, which are called caustic points of the first and second order.

At a *caustic point of the first order*,

$$\mathrm{rank}(\mathbf{Q}) = 1, \tag{5.35}$$

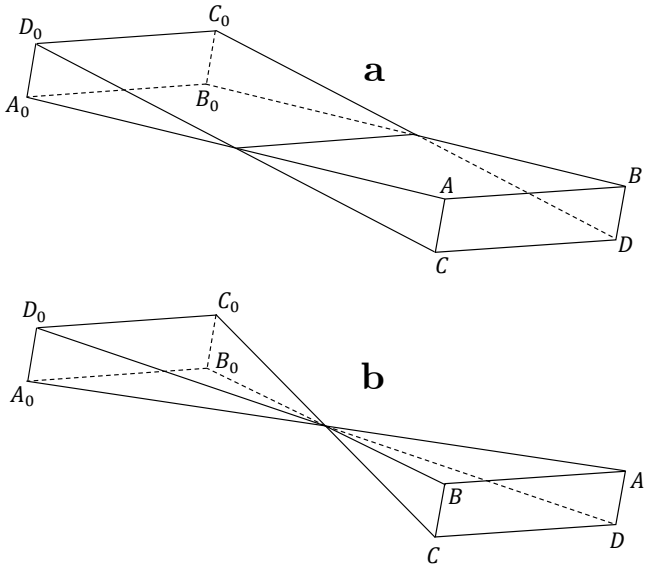


Figure 5.11: Two types of caustic points. At a caustic point of the first order (a), the ray tube reduces to an arc. At a caustic point of the second order (b), the ray tube shrinks to a point.

and the ray tube shrinks to an arc, perpendicular to the direction of propagation. See Figure 5.11(a). At a *caustic point of the second order*,

$$\text{rank}(\mathbf{Q}) = 0, \tag{5.36}$$

and the ray tube shrinks to a point. See Figure 5.11(b).

At caustic points, standard ray theory gives an *infinite amplitude* as the denominator in Equation (5.34) becomes zero, which is not a physical solution. Moreover, when passing through the caustic point of the first order, ray Jacobian J changes sign, and the argument of $J^{1/2}$ takes the phase term $\pm\pi/2$. Similarly, when passing through the caustic point of the second order, the phase term is $\pm\pi$.

The *phase shift due to caustics* is cumulative. The total phase shift when the ray passes through several caustic points is the sum of the individual phase shifts. Consider ray Ω from S to R. The phase shift due to caustics along ray Ω from S to R is given by

$$T^c(R, S) = -\tfrac{1}{2}\pi k(R, S), \tag{5.37}$$

the superscript c denotes that it is induced by caustics. Here $k(R, S)$ is called the *KMAH index* from S to R. In isotropic media, it equals the number of caustic points along ray trajectory Ω from S to R, caustic points of the second order being counted twice. The term KMAH index is introduced by Ziolkowski and Deschamps (Ziolkowski and Deschamps, 1980) acknowledging the work by Keller (Keller, 1958), Maslov (Maslov, 1965), Arnold (Arnold, 1967), and Hörmander (Hörmander, 1971).

The treatment of the infinite amplitude problem will be discussed in detail in Section 5.6.3, while the treatment of phase shifts and the determination of the KMAH index will be discussed in Section 5.6.2.1. Next I shall discuss dynamic ray tracing, namely how the 2×2 matrices $\mathbf{Q}$ and $\mathbf{P}$ are computed along a ray.

5.6.2 Dynamic Ray Tracing

Dynamic ray tracing is the practice of solving a system of several ordinary differential equations along a known ray Ω and yields the first derivatives of position $\vec{x}$ and slowness vector $\vec{p}$ in various coordinate systems (e.g. ray-centered coordinates, ray parameters, Cartesian coordinates) with respect to their initial values. The name is from seismology (Červeny̌ and Hron, 1980), and the term *dynamic* should not be confused with the common use in computer graphics where it usually means the scene is moving.

If I consider a two-parametric orthonormal system of rays, specified by ray parameters γ_1 and γ_2, I can use the dynamic ray tracing system to compute the 2×2 matrices $\mathbf{Q}$ and $\mathbf{P}$, with elements specified in Equation (5.31) along Ω. Note that matrix $\mathbf{Q}$ represents the transformation matrix from ray parameters γ_1 and γ_2 to the ray-centered coordinates q_1 and q_2 and can be used to compute the geometrical spreading. Matrices $\mathbf{Q}$ and $\mathbf{P}$ can be used to compute the 2×2 matrix $\mathbf{M}$ of the second derivative of the travel-time field with respect to q_1 and q_2:

$$\mathbf{M} = \mathbf{P}\mathbf{Q}^{-1}. \tag{5.38}$$

In the following discussion the ray parameters are chosen to be the take-off angles i_0 and ϕ_0 of the rays; see Figure 5.8.

I illustrate the steps of computing $\mathbf{Q}$ and $\mathbf{P}$ from point S to another point R on ray Ω in the following schematic diagram:

$$\begin{array}{ccc} \text{(a) Initial Conditions: } \mathbf{P}(S), \mathbf{Q}(S) & & \mathbf{P}(R), \mathbf{Q}(R) \\ \text{(b) Transform: } \hat{\mathbf{H}}(S) \downarrow & & \uparrow \text{(d) Transform: } \hat{\mathbf{H}}^T(R) \\ \hat{\mathbf{P}}^{(x)}(S), \hat{\mathbf{Q}}^{(x)}(S) & \xrightarrow{\text{(c) Continuation}} & \hat{\mathbf{P}}^{(x)}(R), \hat{\mathbf{Q}}^{(x)}(R) \end{array} \tag{5.39}$$

The important steps are:

(a) First the initial conditions for $\mathbf{Q}$ and $\mathbf{P}$ are given, particularly for the case where S is a point source.

(b) Then matrices $\mathbf{Q}$ and $\mathbf{P}$ are transformed to another coordinate system using a transformation matrix $\hat{\mathbf{H}}$ at point S.

(c) The continuation of the transformed matrices from point S to point R is solved. An analytical solution is given for the special case that I am concerned (Section 5.6.1)

(d) Finally the matrices are transformed back to $\mathbf{Q}$ and $\mathbf{P}$ at point R using the transformation matrix $\hat{\mathbf{H}}^T(R)$

Next I shall elaborate these steps respectively.

(a) Initial conditions for Q and P. If S is a point source, then the matrices $\mathbf{Q}(S)$ and $\mathbf{P}(S)$ are given in the following equations:

$$\mathbf{Q}(S) = \mathbf{0}, \qquad \mathbf{P}(S) = \frac{1}{V(S)} \begin{pmatrix} 1 & 0 \\ 0 & \sin i_0 \end{pmatrix}. \tag{5.40}$$

Here i_0 is the take-off angle between the ray and the x_3-axis; see Figure 5.8.

(b) Transformation matrix $\hat{\mathbf{H}}$ at point S. I would like to transform $\mathbf{Q}$ and $\mathbf{P}$ to the 3×2 matrices $\hat{\mathbf{Q}}^{(x)}$ and $\hat{\mathbf{P}}^{(x)}$, with components:

$$Q_{iJ}^{(x)} = (\partial x_i / \partial \gamma_J)_{\sigma=\text{const.}}, \qquad P_{iJ}^{(x)} = (\partial p_i^{(x)} / \partial \gamma_J)_{\sigma=\text{const.}} \tag{5.41}$$

From Equation (5.29), Equation (5.31), Equation (5.30) and Equation (5.41), it is simple to see that

$$\hat{\mathbf{Q}}^{(x)} = \hat{\mathbf{H}}\mathbf{Q}, \qquad \hat{\mathbf{P}}^{(x)} = \hat{\mathbf{H}}\mathbf{P}. \tag{5.42}$$

Here $\hat{\mathbf{H}}$ is a 3×2 transformation matrix from the ray-centered coordinate system q_1, q_2 to the general Cartesian coordinate system x_1, x_2, x_3:

$$\hat{\mathbf{H}} = \begin{pmatrix} H_{11} & H_{12} \\ H_{21} & H_{22} \\ H_{31} & H_{32} \end{pmatrix} = \begin{pmatrix} e_{11} & e_{21} \\ e_{12} & e_{22} \\ e_{13} & e_{23} \end{pmatrix}. \tag{5.43}$$

e_{1j} and e_{2j} are Cartesian components of the basis vectors $\vec{e}_1$, $\vec{e}_2$ of the Cartesian coordinates.

The two unit vectors, $\vec{e}_1$ and $\vec{e}_2$ can be chosen arbitrarily at the point source S in the plane perpendicular to the ray direction. I chose the following form:

$$\begin{aligned} \vec{e}_1 &\equiv [\cos i_0 \cos \phi_0, \cos i_0 \sin \phi_0, -\sin i_0], \\ \vec{e}_2 &\equiv [-\sin \phi_0, \cos \phi_0, 0]. \end{aligned} \tag{5.44}$$

The direction of $\vec{e}_1$ and $\vec{e}_2$ is demonstrated in Figure 5.8 on a unit sphere with its center at S. $\vec{e}_1$ is oriented along the meridian (constant ϕ_0) and is positive in the direction of positive x_3; $\vec{e}_2$ is oriented along the parallel (constant i_0). Notice that once the ray-centered coordinates has been specified at any reference point of the ray (here at point source S), then they are uniquely determined along the whole ray Ω.

Plugging Equation (5.44) into Equation (5.43) I obtain

$$\hat{\mathbf{H}}(S) = \begin{pmatrix} e_{11} & e_{21} \\ e_{12} & e_{22} \\ e_{13} & e_{23} \end{pmatrix} = \begin{pmatrix} \cos i_0 \cos \phi_0 & -\sin \phi_0 \\ \cos i_0 \sin \phi_0 & \cos \phi_0 \\ -\sin i_0 & 0 \end{pmatrix}. \tag{5.45}$$

Then using Equation (5.42) I am able to find $\hat{\mathbf{Q}}^{(x)}(S)$ and $\hat{\mathbf{P}}^{(x)}(S)$

(c) Continuation of $\hat{\mathbf{Q}}^{(x)}$ and $\hat{\mathbf{P}}^{(x)}$. I would like to determine the 3×2 matrices

$$\hat{\mathbf{Q}}^{(x)} = \begin{pmatrix} Q_{11}^{(x)} & Q_{12}^{(x)} \\ Q_{21}^{(x)} & Q_{22}^{(x)} \\ Q_{31}^{(x)} & Q_{32}^{(x)} \end{pmatrix}, \qquad \hat{\mathbf{P}}^{(x)} = \begin{pmatrix} P_{11}^{(x)} & P_{12}^{(x)} \\ P_{21}^{(x)} & P_{22}^{(x)} \\ P_{31}^{(x)} & P_{32}^{(x)} \end{pmatrix}, \tag{5.46}$$

from one point S to another point R on ray Ω.

The simplest dynamic ray tracing system is obtained for *monotonic parameter* σ along the ray (see Section 5.6.1.1), which can be determined by:

$$\frac{d}{d\sigma} Q_i^{(x)} = P_i^{(x)}, \qquad \frac{d}{d\sigma} P_i^{(x)} = \frac{1}{2} \frac{\partial^2}{\partial x_i \partial x_j} \left(\frac{1}{V^2} \right) Q_j^{(x)}. \tag{5.47}$$

Here I omit the subscript J for γ. A special case that I am concerned about is when V^{-2} is a linear function of coordinates x_i, as shown in Equation (5.27). The dynamic ray tracing system can be simply solved analytically:

$$P_{iJ}^{(x)}(R) = P_{iJ}^{(x)}(S), \qquad Q_{iJ}^{(x)}(R) = Q_{iJ}^{(x)}(S) + \sigma(R,S) P_{iJ}^{(x)}(S), \tag{5.48}$$

where $\sigma(R,S) = \sigma(R) - \sigma(S)$.

Transformation matrix $\hat{\mathbf{H}}$ at point R. I would like to transform the 3×2 matrices $\hat{\mathbf{Q}}^{(x)}(R)$ and $\hat{\mathbf{P}}^{(x)}(R)$ back to the 2×2 matrices $\mathbf{Q}(R)$ and $\mathbf{P}(R)$. Reversing Equation (5.42), the transforms are

$$\mathbf{Q}(R) = \hat{\mathbf{H}}^T(R) \hat{\mathbf{Q}}^{(x)}(R), \qquad \mathbf{P}(R) = \hat{\mathbf{H}}^T(R) \hat{\mathbf{P}}^{(x)}(R), \tag{5.49}$$

since $\hat{\mathbf{H}}$ is an orthonormal transform, $\hat{\mathbf{H}}^{-1} = \hat{\mathbf{H}}^T$. Thus the problem becomes determining $\hat{\mathbf{H}}$ at point R. Remember from Equation (5.29), I have $H_{iJ}(R) = \vec{e}_{Ji}(R)$, so my goal is to find the evolution of the basis vectors $\vec{e}_1$ and $\vec{e}_2$ of the ray-centered coordinates from point S to R.

Within a cell I assume V^{-2} has a constant gradient $\vec{A}$ (5.27). Taking the cross product of Equation (5.28) and the gradient vector $\vec{A}$ I can see that the ray, which is a parabolic curve, completely lies in a plane whose normal is defined by $\vec{p}_0 \times \vec{A}$. I call this plane $\Sigma_{/\!/}$; see Figure 5.12.

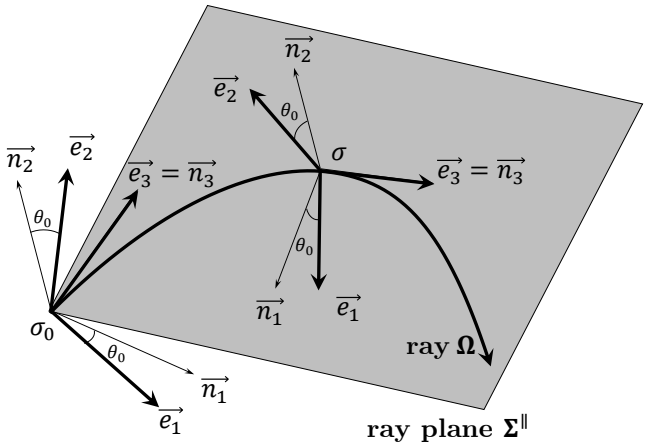


Figure 5.12: Computing $\hat{\mathbf{H}}$ along the ray Ω. $\hat{\mathbf{H}}$ is determined by the basis vectors $\vec{e}_1$, $\vec{e}_2$, and $\vec{e}_3$ of the ray-centered coordinate system. For a ray lying on plane $\Sigma_{/\!/}$, I may define a set of unit vectors $\vec{n}_1$, $\vec{n}_2$, $\vec{n}_3 = \vec{t}$. $\vec{n}_2$ is chosen to be perpendicular to $\Sigma_{/\!/}$. The evolution of $\vec{e}_i$ follows $\vec{n}_i$, where the angle θ_0 between $\vec{e}_1$ and $\vec{n}_1$ (which is also the same between $\vec{e}_2$ and $\vec{n}_2$) is kept fixed.

A set of unit vectors $\vec{n}_1$, $\vec{n}_2$, $\vec{n}_3 = \vec{t}$ orthonormal with each other can be defined with respect to ray Ω and plane $\Sigma_{/\!/}$. I define $\vec{n}_3$ to be tangent to the ray curve

$$\vec{n}_3(\sigma) = \vec{t} = V(\sigma)\vec{p}(\sigma) \tag{5.50}$$

and select $\vec{n}_2$ to be perpendicular to $\Sigma_{/\!/}$, then $\vec{n}_1$ is defined by $\vec{n}_1 = \vec{n}_2 \times \vec{n}_3$. $\vec{n}_2$ does not change in this cell:

$$\vec{n}_2(\sigma) = \vec{n}_2(\sigma_0), \tag{5.51}$$

where σ_0 is the value of σ when entering this cell. See Figure 5.12.

If V^{-2} has a constant gradient, then from Equation (5.28 the slowness vector is

$$\vec{p}(\sigma) = \vec{p}(\sigma_0) + \tfrac{1}{2}\vec{A}(\sigma - \sigma_0). \tag{5.52}$$

Then

$$\begin{aligned}\vec{n}_1(\sigma) &= \vec{n}_2(\sigma) \times \vec{n}_3(\sigma) \\ &= \vec{n}_2(\sigma) \times V(\sigma)\vec{p}(\sigma) \\ &= \vec{n}_2(\sigma_0) \times V(\sigma)\left(\vec{p}(\sigma_0) + \tfrac{1}{2}\vec{A}(\sigma - \sigma_0)\right).\end{aligned} \tag{5.53}$$

Thus $\vec{e}_1(\sigma)$, $\vec{e}_2(\sigma)$ can be determined from $\vec{e}_1(\sigma_0)$, $\vec{e}_2(\sigma_0)$ and the evolution of $\vec{n}_1$, $\vec{n}_2$ from σ to σ_0:

$$\begin{aligned}\vec{e}_1(\sigma) &= \left[\vec{e}_1(\sigma_0) \cdot \vec{n}_1(\sigma_0)\right]\vec{n}_1(\sigma) + \left[\vec{e}_1(\sigma_0) \cdot \vec{n}_2(\sigma_0)\right]\vec{n}_2(\sigma), \\ \vec{e}_2(\sigma) &= \left[\vec{e}_2(\sigma_0) \cdot \vec{n}_1(\sigma_0)\right]\vec{n}_1(\sigma) + \left[\vec{e}_2(\sigma_0) \cdot \vec{n}_2(\sigma_0)\right]\vec{n}_2(\sigma)\end{aligned} \tag{5.54}$$

For point R within this cell, $\hat{\mathbf{H}}(R)$ can be found by plugging $\sigma(R)$ into Equation (5.54) and Equation (5.43), and $\mathbf{Q}(R)$ and $\mathbf{P}(R)$ can be found by Equation (5.49).

5.6.2.1 Phase Shift due to Caustics

The computation of $\mathbf{Q}$ and $\mathbf{P}$ allows us to compute the ray amplitudes using Equation (5.34) and $J = \det \mathbf{Q}$. Moreover, it allows us to determine the argument of $J^{1/2}$ due to phase shifts.

If I discard the parameter s and denote the point in space at s_0 as S and point in space at s as R, then Equation (5.34) can be rewritten as

$$P(R) = \left[\frac{\rho(R)c(R)J(S)}{\rho(S)c(S)J(R)}\right]^{1/2} P(S). \tag{5.55}$$

Alternatively,

$$P(R) = \left[\frac{\rho(R)c(R)}{\rho(S)c(S)}\right]^{1/2} \frac{\mathcal{L}(S)}{\mathcal{L}(\mathcal{R})} \exp\left[\mathrm{i}T^c(R,S)\right] P(S), \tag{5.56}$$

where $T^c(R,S)$ is the phase shift due to caustics, and $\mathcal{L}$ is the geometrical spreading, $\mathcal{L} = |J|^{1/2}$.

In order to compute the phase shift due to caustics $T^c(R,S)$, I have to compute the KMAH index $k(R,S)$ in Equation (5.37). It can be determined by examining the 2×2 transformation matrix $\mathbf{Q}$ from ray parameters γ_1, γ_2 to ray-centered coordinates q_1, q_2. Since $\mathbf{Q}$ can be computed at all points of Ω, the caustic points of the first and second order can be located at points which $\det \mathbf{Q} = 0$, satisfying Equation (5.35) or Equation (5.36.

Consider two consecutive points O^1 and O^2 on ray Ω, where the 2×2 matrix $\mathbf{Q}$ takes values $\mathbf{Q}^1 = \mathbf{Q}(O^1)$ and $\mathbf{Q}^2 = \mathbf{Q}(O^2)$. The following two criteria can be used to determine whether there is a caustic point on Ω between O^1 and O^2.

a. If

$$\det \mathbf{Q}^1 \det \mathbf{Q}^2 < 0, \tag{5.57}$$

there is a caustic point of the first order between O^1 and O^2.

b. Otherwise, if

$$\operatorname{tr}\left[\mathbf{Q}^1(\mathbf{Q}^2)^{-1}\right] \det \mathbf{Q}^1 \det \mathbf{Q}^2 < 0, \tag{5.58}$$

there is a caustic point of the second order between O^1 and O^2. This can be written in a form more useful in programming (Červený et al., 1988):

$$\left(Q^1_{11}Q^2_{22} - Q^1_{12}Q^2_{21} + Q^1_{22}Q^2_{11} - Q^1_{21}Q^2_{12}\right) \det \mathbf{Q}^1 < 0. \tag{5.59}$$

5.6.2.2 Ray Amplitudes

Having computed $\mathbf{Q}$ and $T^c(R, S)$, then the pressure amplitudes on a ray can be computed using (5.56) and $\mathcal{L} = |J|^{1/2} = |\det \mathbf{Q}|^{1/2}$. The only caveat is that for a point source, geometrical spreading $\mathcal{L}(S)$ vanishes at initial point S on ray Ω, and I need to specify a finite $P^0(S)$ at S. By taking

$$\lim_{S' \to S} \{\mathcal{L}(S')P(S')\} = P^0(S), \tag{5.60}$$

where point S' is on ray Ω, I obtain the final equation for ray amplitudes:

$$P^{ray}(R) = \left[\frac{\rho(R)c(R)}{\rho(S)c(S)}\right]^{1/2} \frac{\exp\left[\mathrm{i}T^c(R, S)\right]}{\mathcal{L}(\mathcal{R})} P^0(S). \tag{5.61}$$

5.6.3 Gaussian Beams

In previous sections, I construct the approximate high-frequency solutions of the acoustic wave equation valid on rays. In this section, I shall extend the solutions so that they not only are approximately valid along rays but also in the vicinity of these rays. These elementary solutions,

connected with the individual rays, can be used in the superposition integrals to obtain more general solutions of the acoustic wave equation. The summation of Gaussian beams passing in the vicinity of the receiver, multiplied by some weighting functions, removes certain singularities of the standard ray method (e.g. caustics).

Consider a point R situated on ray Ω, and a point R' situated in the vicinity of R, possibly not on Ω. Then the approximated pressure p^{app}at R' is given by the relation:

$$p^{\text{app}}(R') = P^{ray}(R) \exp\left[-\mathrm{i}\omega(t - T(R' - R))\right]. \tag{5.62}$$

The amplitude $P^{ray}(R)$ is given in Equation (5.61). The travel-time function $T(R', R)$ represents the approximated travel time at R', expressed in terms of the travel time at R. In the ray-centered coordinates system q_1, q_2, $T(R, R')$ reads:

$$T(R', R) = T(R) + \tfrac{1}{2}\mathbf{q}^T(R')\mathbf{M}(R)\mathbf{q}(R'). \tag{5.63}$$

Here $\mathbf{q} = (q_1, q_2)^T$ and $\mathbf{M}$ is the 2×2 matrix of the second derivatives of the travel-time field with respect to ray-centered coordinates q_1, q_2; see Equation (5.38.

The approximate high-frequency solution (Equation (5.62)) of the acoustic wave equation can be generalized by allowing solutions $\mathbf{Q}$ and $\mathbf{P}$ (and therefore $\mathbf{M} = \mathbf{P}\mathbf{Q}^{-1}$ and $\det \mathbf{Q}$ of the dynamic ray tracing system to take complex values. Thus,

$$\mathbf{M} = \mathrm{Re}(\mathbf{M}) + \mathrm{i}\ \mathrm{Im}(\mathbf{M}). \tag{5.64}$$

Assuming that $\mathrm{Im}(\mathbf{M})$ is positive definite, then Equation (5.62) and Equation (5.63) becomes

$$\begin{aligned} p^{\text{beam}}(R') =& P^{ray}(R) \exp\left[-\mathrm{i}\omega(t - T(R) - \tfrac{1}{2}\mathbf{q}^T(R')\mathbf{M}(R)\mathbf{q}(R'))\right] \\ =& P^{ray}(R) \exp\left[-\mathrm{i}\omega(t - T(R) - \tfrac{1}{2}\mathbf{q}^T(R')\mathrm{Re}(\mathbf{M}(R))\mathbf{q}(R'))\right] \\ & \times \exp\left[-\tfrac{1}{2}\omega\mathbf{q}^T(R')\mathrm{Im}(\mathbf{M}(R))\mathbf{q}(R')\right]. \end{aligned} \tag{5.65}$$

The solution has an amplitude profile closely concentrated about the central ray and represents a beam. As can be seen in the last term of Equation (5.65), the amplitude extends to the vicinity of

ray Ω with non-zero $\mathbf{q}$ with a profile of a Gaussian function. This is why solutions as defined in Equation (5.65) with $\mathrm{Im}(\mathbf{M}(R)) \neq \mathbf{0}$ are called *Gaussian beams*. Complex-valued matrices $\mathbf{M}$ and $\mathbf{Q}$ must satisfy three conditions

a. $\mathbf{Q}$ is regular, i.e. $\det(\mathbf{Q}) \neq 0$ and $\det(\mathbf{M}) \neq \infty$.

b. $\mathbf{M}$ is symmetrical.

c. $\mathrm{Im}(\mathbf{M})$ is positive definite.

5.6.4 Summation Methods

Just like the spherical wave in a homogeneous medium can be expressed as the superposition of the plane waves using the classical Weyl integral (Weyl, 1919), it is possible to construct useful expressions for the wave field by integral superposition of asymptotic ray-based solutions. These superposition integrals sum up individual contributions of Gaussian beams and are not exact. But they provide a uniform asymptotic solution of the acoustic wave equation, valid even in certain singular regions of the ray method.

Consider an acoustic wave propagating in a inhomogeneous medium and the relevant orthonormal system of rays $\Omega(\gamma_1, \gamma_2)$, parameterized by two ray parameters γ_1 and γ_2. On each ray, I specify one initial point S_γ, at which some initial conditions are specified. I assume that the 2×2 matrices $\mathbf{Q}^a(S_\gamma)$, $\mathbf{P}^a(S_\gamma)$, and $\mathbf{M}^a(S_\gamma) = \mathbf{P}^a(S_\gamma)\mathbf{Q}^{a-1}(S_\gamma)$, corresponding to the actual ray field $\Omega(\gamma_1, \gamma_2)$, are known at S_γ. The superscript "a" is used to emphasize that these matrices correspond to the actual ray field. These matrices are fixed for the acoustic wave under consideration. They should be distinguished from the 2×2 complex-valued symmetric matrix $\mathbf{M}(S_\gamma)$ used to describe Gaussian Beams, which should be specified in some other way. See Section 5.6.4.4.

5.6.4.1 Superposition Integrals

I would like to determine the wavefield of the acoustic wave $p(R, \omega)$ at a fixed receiver R. I do not have to identify the ray that exactly passes through R. Instead, the wavefield at R is calclulated by a weighted superposition of Gaussian beams connected with rays $\Omega(\gamma_1, \gamma_2)$ passing in the vicinity of R. See Figure 5.13.

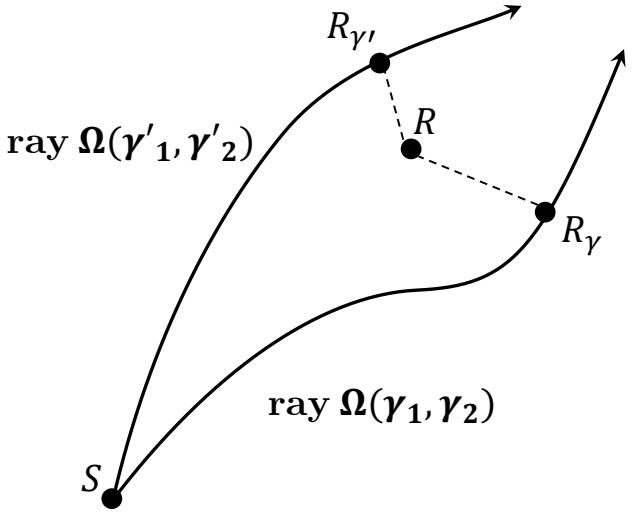


Figure 5.13: Approximation of the wave field at R as a weighted sum of contributions from nearby Gaussian beams. Two Gaussian beams connected to ray $\Omega(\gamma_1, \gamma_2)$ and $\Omega(\gamma'_1, \gamma'_2)$ are shown, where points R_γ and $R_{\gamma'}$ close to R (not necessarily the closest) are situated.

In the frequency domain (neglecting the $\exp[-\mathrm{i}\omega t]$ factor), the superposition integral reads as

$$p(R, \omega) = \iint_{\mathcal{D}} \Phi(\gamma_1, \gamma_2) P^{ray}(R_\gamma) \exp[\mathrm{i}\omega T(R, R_\gamma)] \, \mathrm{d}\gamma_1 \, \mathrm{d}\gamma_2. \tag{5.66}$$

The integral is over the rays specified by ray parameters γ_1 and γ_2; $\mathcal{D}$ denotes the region of ray parameters under consideration. Function $\Phi(\gamma_1, \gamma_2)$ is the weighting function, which will be determined in Section 5.6.4.2. Point R_γ is situated on the same ray $\Omega(\gamma_1, \gamma_2)$ as S_γ, and should be chosen as close to the fixed point R as possible. The function P^{ray} represents the pressure amplitude computed by Equation (5.61) and may be complex-valued. The travel-time function $T(R, R\gamma)$ represents the travel time at R, calculated by approximating from the travel time $T(R_\gamma)$ at R_γ situated on a near-by ray $\Omega(\gamma_1, \gamma_2)$; it will be discussed in Section 5.6.4.3.

5.6.4.2 Determination of the Weighting Function

The weighting function $\Phi(\gamma_1, \gamma_2)$ is determined by matching the approximate superposition integral to a known standard ray-theory solution at point R in a regular ray region (i.e. no singularities).

I shall not go into details here but merely presents the result. I refer the interested reader to Červený (Červený, 2005).

The final expression of the weighting function $\Phi(\gamma, \gamma_2)$ is given as follows:

$$\Phi(\gamma_1, \gamma_2) = (\omega/2\pi)\left[-\det \boldsymbol{\mathcal{M}}(R_\gamma)\right]^{1/2} \left|\det \mathbf{Q}^a(R_\gamma)\right|. \tag{5.67}$$

Here the 2×2 matrix $\boldsymbol{\mathcal{M}}(R_\gamma)$ is defined as

$$\boldsymbol{\mathcal{M}}(R_\gamma) = \mathbf{M}(R_\gamma) - \mathbf{M}^a(R_\gamma). \tag{5.68}$$

The argument of $[-\det \boldsymbol{\mathcal{M}}(R_\gamma)]^{1/2}$ is given by the following relation for $\mathbf{W}$ a constant 2×2 matrix with $\det \mathbf{W} \neq 0$:

$$\begin{aligned} &\mathrm{Re}[-\det \mathbf{W}]^{1/2} > 0 && \text{for Im } \mathbf{W} \neq \mathbf{0}, \\ &[-\det \mathbf{W}]^{1/2} = |\det \mathbf{W}|^{1/2} \exp\left[-\mathrm{i}\tfrac{\pi}{4}\mathrm{Sgn}\mathbf{W}\right] && \text{for Im } \mathbf{W} = \mathbf{0}. \end{aligned} \tag{5.69}$$

Sgn $\mathbf{W}$ denotes the signature of the real-valued matrix $\mathbf{W}$; it equals the the number of its positive eigenvalues minus the number of its negative eigenvalues. Thus, it takes on values of 2, 0, or -2.

5.6.4.3 Travel-Time Function

Function $T(R, R_\gamma)$ represents the travel time at R, approximated as the second order Taylor expansion of the travel time around R_γ on ray Ω. R_γ may be chosen arbitrarily on rays Ω, but close to R. The only requirement is that the distance $\left|\vec{x}(R) - \vec{x}(R_\gamma)\right|$ is small and the terms higher than quadratic may be neglected. Denote the Cartesian coordinates of points R and R_γ by $x_i(R)$ and $x_i(R_\gamma)$, and introduce $x_i(R, R_\gamma) = x_i(R) - x_i(R_\gamma)$. Then the quadratic expansion in terms of $x_i(R, R_\gamma)$ is as follows:

$$T(R, R_\gamma) = T(R_\gamma) + x_i(R, R_\gamma) p_i^{(x)}(R_\gamma) + \tfrac{1}{2} x_i(R, R_\gamma) x_j(R, R_\gamma) M_{ij}^{(x)}(R_\gamma). \tag{5.70}$$

$T(R_\gamma)$ is the travel time for point R_γ on ray Ω, which can be computed using cell methods segment-by-segment with Equation (5.28).

$M_{ij}^{(x)}$ in Equation (5.70) are the elements in the 3×3 matrix $\hat{\mathbf{M}}^{(x)}$:

$$\hat{\mathbf{M}}^{(x)}(R_\gamma) = \hat{\mathbf{H}}(R_\gamma) \begin{pmatrix} & \mathbf{M}(R_\gamma) & & M_{13}(R_\gamma) \\ & & & M_{23}(R_\gamma) \\ M_{13}(R_\gamma) & M_{23}(R_\gamma) & & M_{33}(R_\gamma) \end{pmatrix} \hat{\mathbf{H}}^T(R_\gamma). \tag{5.71}$$

Here $\hat{\mathbf{H}}$ is the 3×3 transformation matrix from the ray-centered coordinates to the Cartesian coordinates, which is defined in Equation (5.29). The 2×2 matrix $\mathbf{M}(R_\gamma)$ in Equation (5.71) is free and may be chosen in various ways. See Section 5.6.4.4. The other elements are

$$\begin{aligned} M_{13}(R_\gamma) &= -(v^{-2}v_{,1})_{R_\gamma}, \\ M_{23}(R_\gamma) &= -(v^{-2}v_{,2})_{R_\gamma}, \\ M_{33}(R_\gamma) &= -(v^{-2}v_{,3})_{R_\gamma}. \end{aligned} \tag{5.72}$$

Here

$$\begin{aligned} v &= \left[V(q_1, q_2, s)\right]_{q_1=q_2=0, s=s(R_\gamma)}, \\ v_{,i} &= \left[\partial V(q_1, q_2, s)/\partial q_i\right]_{q_1=q_2=0, s=s(R_\gamma)}. \end{aligned} \tag{5.73}$$

Computing $v_{,i}$ is easy, notice that

$$v_{,i} = \partial V/\partial q_i = H_{ki}\partial V/\partial x_k. \tag{5.74}$$

In a cell with constant gradient of V^{-2}, $\partial V/\partial x_k$ can be analytically solved by taking derivatives of Equation (5.27),

$$\partial V^{-2}/\partial x_k = -2V^{-3}\partial V/\partial x_k = A_k, \tag{5.75}$$

thus

$$\partial V/\partial x_k = -\tfrac{1}{2}V^3 A_k. \tag{5.76}$$

Combining Equation (5.70) through Equation (5.76), I am able to compute the travel-time function $T(R, R_\gamma)$ in Equation (5.66).

5.6.4.4 Specification of Matrix M

Superposition integral (Equation (5.66)) is influenced by the choice of the 2×2 matrix $\mathbf{M}$. It is common to specify $\mathbf{M}$ at points R_γ. The physical meaning of $\mathrm{Re}(\mathbf{M}(R_\gamma))$ is the *geometrical properties of the wavefront* of the Gaussian Beam. Because $\mathrm{Re}\mathbf{M}(R_\gamma)$ is always symmetrical, its eigenvalues are always real. The eigenvalues times the speed V represent the principal curvatures of the wavefront of the Gaussian beam. Also $\mathrm{Im}(\mathbf{M}(R_\gamma))$ determines the amplitude profile of the Gaussian beams. Therefore, I may consider expanding the wave field into locally plane waves with a Gaussain amplitude windowing by using $\mathrm{Re}(\mathbf{M}(R_\gamma)) = \mathbf{0}$ and $\mathrm{Im}(\mathbf{M}(R_\gamma))$ positive definite.

In general, I can choose a positive-definite 2×2 matrix $\mathrm{Im}(\mathbf{M}(R_\gamma))$ arbitrarily, which controls the width of Gaussian beams under consideration. There are options that can minimize the error of computations, and options that can suppress the quadratic terms from the expansion of $\mathrm{Re}(T(R, R_\gamma))$. I shall not discuss the problem of choice of $\mathbf{M}(R_\gamma)$ in details. For more details, see Červený (Červený, 1985) and Klimeš (KlimeÅ and PÅenk, 1989).

5.6.4.5 Summation Methods: Discussion

The final form of the superposition integral is as follows:

$$p(R,\omega) = \frac{\omega}{2\pi} \iint_{\mathcal{D}} P^{ray}(R_\gamma) \left[-\det \boldsymbol{\mathcal{M}}(R_\gamma) \right]^{1/2} \times |\det \boldsymbol{Q}^a(R_\gamma)| \exp[\mathrm{i}\omega T(R, R_\gamma)] \, \mathrm{d}\gamma_1 \, \mathrm{d}\gamma_2. \tag{5.77}$$

When programming the computation, a simple alternative version of the superposition integral (Equation 5.77)) can be used:

$$p(R,\omega) = \frac{\omega}{2\pi} \iint_{\mathcal{D}} P^{ray}(R_\gamma) \left[-\det \boldsymbol{\mathcal{N}}(R_\gamma) \right]^{1/2} \exp[\mathrm{i}\omega T(R, R_\gamma)] \, \mathrm{d}\gamma_1 \, \mathrm{d}\gamma_2, \tag{5.78}$$

where the 2×2 matrix $\boldsymbol{\mathcal{N}}(R_\gamma)$ is given by the relation:

$$\boldsymbol{\mathcal{N}}(R_\gamma) = \mathbf{Q}^{aT}(\mathbf{M} - \mathbf{M}^a)\mathbf{Q}^a = -\mathbf{Q}^{aT}\mathbf{P}^a + \mathbf{Q}^{aT}\mathbf{M}\mathbf{Q}^a. \tag{5.79}$$

All the quantities are taken at R_γ. The argument of $[-\det \boldsymbol{\mathcal{N}}(R_\gamma)]^{1/2}$ is again given by Equation (5.69), and the travel-time function $T(R, R_\gamma)$ is given by Equation (5.70). Pressure amplitude $P^{ray}(R_\gamma)$ can be computed by Equation (5.61), where $\mathcal{L}$ computed by $|\det \mathbf{Q}|$ and phase shift T^c given by Equation (5.37), computed as discussed in Section 5.6.2.1.

The main disadvantage of the Gaussian beam summation solution is that it depends on the free parameters (i.e. on the widths of the Gaussian beams) in singular regions. In the vicinity of caustic, broad Gaussian beams (small Im$\mathbf{M}$) are desired; in some other cases like computing edge diffractions, very narrow Gaussian beams are required. The optimum choice of Im$\mathbf{M}$ that suits for every case is not known and requires further research.

CHAPTER 6: CONCLUSION AND FUTURE WORK

The contribution of my dissertation lies in providing adaptive modeling of detail for the three problems related to physically-based sound simulation, namely, Liquid Sounds, Rigid Body Sounds, and Sound Propagation. In the area of liquid sounds, I have presented different techniques for synthesizing liquid sounds depending on the level of detail of how bubbles are modeled, thus enabling the control over the trade-off between realsim and computational cost. The system that I have developed has been integrated with a real-time shallow-water fluid simulator and a full 3D grid-SPH fluid simulator, to generate rich liquid sounds automatically.

The second part of my work is on improving the realism of rigid body sounds. First, I proposed using prerecorded audio clips to estimate material parameters that capture the inherent quality of the recorded material. Based on psychoacoustic principles, these estimated parameters allows linear modal synthesis to generate sound that bears a perceptual similarity to the example recording on the first level. On the second level, details from the example recording that are not captured by the linear modal model are computed, transferred, and compensated in the final synthesized sound. We have demonstrated the effectiveness of the system by estimating material parameters and residuals from various objects of different materials and applying them on virtual objects of different geometries to generate rich and complex contact sounds.

Finally, I have developed a hybrid sound propagation method that combines geometric and numerical acoustic techniques. In regions far away from objects, sound propagation is modeled by the more efficient ray-based, geometric technique. Then in limited regions near objects, wave phenomena are modeled using the more accurate and costly numerical technique. This approach allows allocating the computation resources on where it matters the most and is able to handle sound propagation for large, indoor and outdoor complex scenes that are previously infeasible to simulate accurately. I also discuss the extension of the geometric acoustics part to handle propagation in inhomogeneous medium, the challenges that come with it, and how to overcome them.

Future Work: For each of the techniques that I have described in this thesis, there are many possible improvements to be made and many future directions worth investigating, and I have described them individually in the previous chapters. Here I would like to discuss the general research trend for future in a larger scope.

Computer graphics has seen tremendous development in the past few decades. Many sub-areas of computer graphics have benefitted from *physics simulation*, such as physically-based rendering techniques and physically-based animation of fluid, rigid and deformable bodies, characters, etc. These techniques have enabled stunning visual renderings in many different applications including games, movies, and virtual reality. Can physically-based sound simulation achieve the same level of maturity and wide application as its visual counterpart? In theory it should. Just as physics determines how light travels in space and how objects deform and move, physics dictates how sound is generated and propagates, and simulating the physics of sound should be an equally powerful tool for generating realistic sound effects. But there are several challenges to be overcome.

One challenge is to improve the quality. While visual simulation has already been able to produce images and animations so real that human eyes cannot tell whether they are computer-generated or not, digitally-synthesized sounds still sound a little 'artificial' to human ears. One reason is that the physical models that we used for sound simulation are not complete. For example, the Rayleigh Damping model, which is widely used for simulating rigid body sounds, cannot describe all types of materials– in fact, no one existing damping model can. When the model is not complete to allow a *forward* synthesis of sounds, operating on recorded sounds and modifying them according to the needs is another option. My work on example-guided modal synthesis follows this direction, and the residuals are used to capture the difference between the recorded sounds and the model-synthesized sounds. However, our residual transfer algorithm is still a heuristic, and a better understanding of the the source and mechanism of the residuals can lead to a better transfer algorithm. Similarly, more complete models must be used for sound propagation. For example in the case of outdoor acoustics, wind, turbulence, temperature gradient, and many complicated physical processes all affect what we hear in the end, and the sound propagation model should consider all these to produce realistic acoustic effects.

Another challenge is to improve efficiency. This aspect involves developing better computational techniques as well as perceptual approximations. For example, in recent years more and more gain in

computing power comes from all kinds of parallelism, from CPU to GPU to cloud computing. Parallel algorithms need to be designed and developed to fully utilize the computing power. Also, more gross approximation and more aggressive simplification, perhaps based on better understandings of psychoacoustics, need to be continuously investigated. For example, accurate sound propagation is in many ways analogous to global illumination in visual rendering. A whole range of approximation techniques such as ambient occlusion have been developed for visual rendering for interactive applications, can we develop something similar in effect for sound rendering?

My work on adaptive modeling of details aims to balance the quality and efficiency of sound simulation techniques. The proposed algorithms provide two to three levels of details that can be chosen by the user. In the future more levels can be added on both ends to handle a wider range of applications– more sophisticated models that are able to generate more realistic sounds on one end, and more crude approximations that allow faster computation on the other end. Take the liquid sound simulation for example. Currently the highest level decomposes bubbles to spherical harmonics, which is limited to star-shaped bubbles, and we still treat each bubble independently from other bubbles. In the future we could add a more general model for bubbles of arbitrary shapes having complex interactions (popping, merging, acoustic-coupling, etc.) Similarly, the lowest level considers only the properties of the surface and the statistical distribution of bubbles, but we still simulate one sine wave for each bubble. And therefore it is still challenging to simulate sounds for large-scale fluid motion like a flooding city or a waterfall (whose visual simulation are already possible), where billions of bubbles emit sounds simultaneously. However in such scenes the final sound poses a noise-like quality, and it might be more efficient to model the sound as a noise texture and apply modifications in the spectral domain. It is an interesting research direction to explore more choices of different level-of-detail modeling and how to combine them seamlessly for each application.

I also hope to see exploration of the space of sound effects that can be simulated. For example, the synthesis of sounds of floors creaking, bottles buckling, papers crumpling and tearing, and shock wave sounds such as explosion and thunder. The propagation of the shock wave sounds needs nonlinear wave equation which is still an active research in the physics and acoustics community. In computer graphics, almost all natural phenomena and physical interactions can be visually simulated, at least to an extent of perceptual plausibility. Sound simulation has to cover a larger base than it currently has to be widely used in graphics applications.

I hope that in the future more researchers will devote themselves into advancing sound simulation techniques and developing more tools, so that physically-based sound simulation will be used more widely in many different applications.

BIBLIOGRAPHY

Adams, B., Pauly, M., Keiser, R., and Guibas, L. J. (2007). Adaptively sampled particle fluids. In *ACM SIGGRAPH 2007 papers*, page 48, San Diego, California. ACM.

Adrien, J.-M. (1991). Representations of musical signals. chapter The missing link: modal synthesis, pages 269–298. MIT Press, Cambridge, MA, USA.

Alarcao, D., Santos, D., and Coelho, L. B. (2010). Virtusound – a real-time auralization system. In *Proc. International Congress on Acoustics*.

Antani, L., Chandak, A., Savioja, L., and Manocha, D. (2012). Interactive sound propagation using compact acoustic transfer operators. *ACM Trans. Graph.*, 31(1):7:1–7:12.

Aretz, M. (2012). *Combined wave and ray based room acoustic simulations of small rooms: challenges and limitations on the way to realistic simulation results*. PhD thesis, Aachener Beitrge zur Technischen Akustik.

Arfken, G. B., Weber, H. J., and Ruby, L. (1985). *Mathematical methods for physicists*, volume 3. Academic press San Diego.

Arnold, V. (1967). Characteristic class entering in quantization conditions. *Funct. Anal. Appl*, 1:1–13.

Audiokinetic (2011). Wwise SoundSeed Impact. `http://www.audiokinetic.com/en/products/wwise-add-ons/soundseed/introduction`.

Barbone, P. E., Montgomery, J. M., Michael, O., and Harari, I. (1998). Scattering by a hybrid asymptotic/finite element method. *Computer methods in applied mechanics and engineering*, 164(1):141–156.

Batty, C., Bertails, F., and Bridson, R. (2007). A fast variational framework for accurate solid-fluid coupling. *ACM Trans. Graph.*, 26(3):100.

Ben-Artzi, A., Egan, K., Durand, F., and Ramamoorthi, R. (2008). A precomputed polynomial representation for interactive BRDF editing with global illumination. *ACM Transactions on Graphics (TOG)*, 27(2):13.

Besl, P. J. and McKay, N. D. (1992). A method for registration of 3-D shapes. *IEEE Transactions on pattern analysis and machine intelligence*, pages 239–256.

Blauert, J. (1983). *Spatial hearing: The psychophysics of human sound localization*. MIT Press (Cambridge, Mass.).

Bleistein, N. (1984). *Mathematical methods for wave phenomena*. Academic Press, New York.

Bonneel, N., Drettakis, G., Tsingos, N., Viaud-Delmon, I., and James, D. (2008). Fast modal sounds with scalable frequency-domain synthesis. *ACM Transactions on Graphics (TOG)*, 27(3):24.

Borish, J. (1984). Extension of the image model to arbitrary polyhedra. *The Journal of the Acoustical Society of America*, 75:1827.

Botteldooren, D. (1994). Acoustical finite-difference time-domain simulation in a quasi-cartesian grid. *The Journal of the Acoustical Society of America*, 95(5):2313–2319.

Botteldooren, D. (1995). Finite-difference time-domain simulation of low-frequency room acoustic problems. *Acoustical Society of America Journal*, 98:3302–3308.

Bridson, R. and Müller-Fischer, M. (2007). Fluid simulation: SIGGRAPH 2007 course notes. In *ACM SIGGRAPH 2007 courses*, pages 1–81, San Diego, California. ACM.

Carlson, M., Mucha, P. J., and Turk, G. (2004). Rigid fluid: animating the interplay between rigid bodies and fluid. In *ACM SIGGRAPH 2004 Papers*, pages 377–384, Los Angeles, California. ACM.

Červenỳ, V. and Hron, F. (1980). The ray series method and dynamic ray tracing system for three-dimensional inhomogeneous media. *Bulletin of the Seismological Society of America*, 70(1):47–77.

Červenỳ, V., Popov, M. M., and Pšenčík, I. (1982). Computation of wave fields in inhomogeneous mediagaussian beam approach. *Geophysical Journal International*, 70(1):109–128.

Chadwick, J. N., An, S. S., and James, D. L. (2009). Harmonic shells: a practical nonlinear sound model for near-rigid thin shells. In *SIGGRAPH Asia '09: ACM SIGGRAPH Asia 2009 papers*, pages 1–10, New York, NY, USA. ACM.

Chadwick, J. N. and James, D. L. (2011). Animating fire with sound. In *ACM Transactions on Graphics (TOG)*, volume 30, page 84. ACM.

Chandak, A., Lauterbach, C., Taylor, M., Ren, Z., and Manocha, D. (2008). Ad-frustum: Adaptive frustum tracing for interactive sound propagation. *IEEE Trans. Visualization and Computer Graphics*, 14(6):1707–1722.

Cheng, A. and Cheng, D. (2005). Heritage and early history of the boundary element method. *Engineering Analysis with Boundary Elements*, 29(3):268–302.

Cook, P. R. (1996). Physically informed sonic modeling (PhISM): percussive synthesis. In *Proceedings of the 1996 International Computer Music Conference*, pages 228–231. The International Computer Music Association.

Cook, P. R. (1997). Physically informed sonic modeling (phism): Synthesis of percussive sounds. *Computer Music Journal*, 21(3):38–49.

Cook, P. R. (2002). *Real Sound Synthesis for Interactive Applications*. A. K. Peters, Ltd., Natick, MA, USA.

Cook, P. R. and Scavone, G. P. (2010). The synthesis ToolKit in c++ 4.3.1.

Corbett, R., van den Doel, K., Lloyd, J. E., and Heidrich, W. (2007). Timbrefields: 3d interactive sound models for real-time audio. *Presence: Teleoperators and Virtual Environments*, 16(6):643–654.

Deane, G. B. and Stokes, M. D. (2002). Scale dependence of bubble creation mechanisms in breaking waves. *Nature*, 418(6900):839–844.

Ding, J., Tsaur, F. W., Lips, A., and Akay, A. (2007). Acoustical observation of bubble oscillations induced by bubble popping. *Physical Review. E, Statistical, Nonlinear, and Soft Matter Physics*, 75(4 Pt 1).

Dobashi, Y., Yamamoto, T., and Nishita, T. (2003). Real-time rendering of aerodynamic sound using sound textures based on computational fluid dynamics. *ACM Trans. Graph.*, 22(3):732–740.

Dobashi, Y., Yamamoto, T., and Nishita, T. (2004). Synthesizing sound from turbulent field using sound textures for interactive fluid simulation. *Computer Graphics Forum*, 23(3):539–545.

Dubuisson, M. P. and Jain, A. K. (1994). A modified hausdorff distance for object matching. In *Proceedings of 12th International Conference on Pattern Recognition*, volume 1, pages 566–568. IEEE Comput. Soc. Press.

Eyring, C. F. (1930). Reverberation time in "dead" rooms. *The Journal of the Acoustical Society of America*, 1(2A):217–241.

Fedkiw, R., Stam, J., and Jensen, H. W. (2001). Visual simulation of smoke. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 15–22. ACM.

Florens, J. and Cadoz, C. (1991). The physical model: modeling and simulating the instrumental universe. In *Representations of musical signals*, pages 227–268. MIT Press.

Fontana, F. (2003). *The sounding object*. Mondo Estremo.

Foster, N. and Fedkiw, R. (2001). Practical animation of liquids. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 23–30. ACM.

Foster, N. and Metaxas, D. (1996). Realistic animation of liquids. *Graph. Models Image Process.*, 58(5):471–483.

Funkhouser, T., Carlbom, I., Elko, G., Pingali, G., Sondhi, M., and West, J. (1998). A beam tracing approach to acoustic modeling for interactive virtual environments. In *Proc. SIGGRAPH 1998*, pages 21–32.

Funkhouser, T., Tsingos, N., and Jot, J.-M. (2004). Survey of methods for modeling sound propagation in interactive virtual environment systems. *Presence*.

Goldstein, H. (1980). Classical mechanics. *Reading, Mass.: Addison-Wesley*, 1.

Gope, C. and Kehtarnavaz, N. (2007). Affine invariant comparison of point-sets using convex hulls and hausdorff distances. *Pattern recognition*, 40(1):309–320.

Granier, E., Kleiner, M., Dalenbck, B.-I., and Svensson, P. (1996). Experimental auralization of car audio installations. *Journal of the Audio Engineering Society*, 44(10):835–849.

Griffin, D. and Lim, J. (2003). Signal estimation from modified short-time Fourier transform. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 32(2):236–243.

Gumerov, N. A. and Duraiswami, R. (2004). *Fast multipole methods for the Helmholtz equation in three dimensions*. Elsevier Science.

Gumerov, N. A. and Duraiswami, R. (2009). A broadband fast multipole accelerated boundary element method for the three-dimensional helmholtz equation. *J. Acoustical Society of America*, 125(1):191–205.

Hampel, S., Langer, S., and Cisilino, A. P. (2008). Coupling boundary elements to a raytracing procedure. *International journal for numerical methods in engineering*, 73(3):427–445.

Hess, P. (2007). *Extended Boundary Conditions for Shallow Water Simulations*. PhD thesis, ETH Zurich.

Hong, J., Lee, H., Yoon, J., and Kim, C. (2008). Bubbles alive. In *ACM SIGGRAPH 2008 papers*, pages 1–4, Los Angeles, California. ACM.

Hörmander, L. (1971). Fourier integral operators. i. *Acta mathematica*, 127(1):79–183.

Imura, M., Nakano, Y., Yasumuro, Y., Manabe, Y., and Chihara, K. (2007). Real-time generation of CG and sound of liquid with bubble. In *ACM SIGGRAPH 2007 posters*, page 97, San Diego, California. ACM.

ISO (2003). *ISO 226: 2003: AcousticsNormal equalloudness-level contours*. International Organization for Standardization.

James, D., Barbič, J., and Pai, D. (2006a). Precomputed acoustic transfer: output-sensitive, accurate sound generation for geometrically complex vibration sources. In *ACM SIGGRAPH 2006 Papers*, page 995. ACM.

James, D. L., Barbic, J., and Pai, D. K. (2006b). Precomputed acoustic transfer: output-sensitive, accurate sound generation for geometrically complex vibration sources. In *Proc. of ACM SIGGRAPH*, pages 987–995.

Jean, P., Noe, N., and Gaudaire, F. (2008). Calculation of tyre noise radiation with a mixed approach. *Acta Acustica united with Acustica*, 94(1):91–103.

Jensen, F. B., Kuperman, W. A., Porter, M. B., and Schmidt, H. (2011). *Computational ocean acoustics*. Springer Science+Business Media, LLC, New York.

Joslin, C. and Magnenat-Thalmann, N. (2003). Significant facet retrieval for real-time 3d sound rendering in complex virtual environments. In *Proc. ACM Symposium on Virtual Reality Software and Technology*.

Karjalainen, M. and Erkut, C. (2004). Digital waveguides versus finite difference structures: equivalence and mixed modeling. *EURASIP J. Appl. Signal Process.*, 2004(1):978–989.

Keller, J. B. (1958). A geometrical theory of diffraction. In *Calculus of variations and its applications*, volume 8, pages 27–52, New York. McGraw Hill.

Kleiner, M., Dalenbäck, B.-I., and Svensson, P. (1993). Auralization - an overview. *JAES*, 41:861–875.

KlimeÅ, L. and PÅenk, R. I. (1989). Optimization of the shape of gaussian beams of a fixed length. *Studia Geophysica et Geodaetica*, 33(2):146–163.

Kouyoumjian, R. G. and Pathak, P. H. (1974). A uniform geometrical theory of diffraction for an edge in a perfectly conducting surface. *Proc. of IEEE*, 62:1448–1461.

Lagarias, J. C., Reeds, J. A., Wright, M. H., and Wright, P. E. (1999). Convergence properties of the Nelder-Mead simplex method in low dimensions. *SIAM Journal on Optimization*, 9(1):112–147.

Lakatos, S., McAdams, S., and Caussé, R. (1997). The representation of auditory source characteristics: Simple geometric form. *Attention, Perception, & Psychophysics*, 59(8):1180–1190.

Lauterbach, C., Chandak, A., and Manocha, D. (2007). Interactive sound propagation in dynamic scenes using frustum tracing. *IEEE Trans. Visualization and Computer Graphics*, 13(6):1672–1679.

Leighton, T. G. (1994). *The acoustic bubble*.

Lentz, T., Schroeder, D., Vorlander, M., and Assenmacher, I. (2007). Virtual reality system with integrated sound field simulation and reproduction. *EURASIP J. Applied Signal Processing*.

Levine, S. N., Verma, T. S., and Smith, J. O. (1998). Multiresolution sinusoidal modeling for wide-band audio with modifications. In *Acoustics, Speech and Signal Processing, 1998. Proceedings of the 1998 IEEE International Conference on*, volume 6, pages 3585–3588 vol. 6. IEEE.

Lloyd, D. B., Raghuvanshi, N., and Govindaraju, N. K. (2011). Sound Synthesis for Impact Sounds in Video Games. In *Proceedings of Symposium on Interactive 3D Graphics and Games*.

Lokki, T., Southern, A., Siltanen, S., and Savioja, L. (2011). Studies of epidaurus with a hybrid room acoustics modeling method. In *Acoustics of Ancient Theaters Patras, Greece*.

Longuet-Higgins, M. S. (1989a). Monopole emission of sound by asymmetric bubble oscillations. part 1. normal modes. *Journal of Fluid Mechanics*, 201:525–541.

Longuet-Higgins, M. S. (1989b). Monopole emission of sound by asymmetric bubble oscillations. part 2. an initial-value problem. *Journal of Fluid Mechanics*, 201:543–565.

Longuet-Higgins, M. S. (1990). Bubble noise spectra. *The Journal of the Acoustical Society of America*, 87(2):652–661.

Longuet-Higgins, M. S. (1991). Resonance in nonlinear bubble oscillations. *Journal of Fluid Mechanics*, 224:531–549.

Longuet-Higgins, M. S. (1992). Nonlinear damping of bubble oscillations by resonant interaction. *The Journal of the Acoustical Society of America*, 91(3):1414–1422.

Lorensen, W. E. and Cline, H. E. (1987). Marching cubes: A high resolution 3D surface construction algorithm. *SIGGRAPH Comput. Graph.*, 21(4):163–169.

Losasso, F., Gibou, F., and Fedkiw, R. (2004). Simulating water and smoke with an octree data structure. In *ACM SIGGRAPH 2004 Papers*, pages 457–462, Los Angeles, California. ACM.

Ludwig, D. (1966). Uniform asymptotic expansions at a caustic. *Communications on Pure and Applied Mathematics*, 19(2):215–250.

Maslov, V. P. (1965). *Teoriya vozmushchenii i asymptoticheskie metody (Theory of Perturbations and Asymptotic Methods)*. Moscow State University Press, Moscow.

Mehra, R., Raghuvanshi, N., Antani, L., Chandak, A., Curtis, S., and Manocha, D. (2013). Wave-based sound propagation in large open scenes using an equivalent source formulation. *ACM Trans. Graph.*, 32(2):19:1–19:13.

Mihalef, V., Metaxas, D., and Sussman, M. (2009). Simulation of two-phase flow with sub-scale droplet and bubble effects. *Proceedings of Eurographics 2009*, 28.

Minnaert, M. (1933). On musical air bubbles and the sound of running water. *Philosophical Magazine*, 16:235–248.

Morchen, F., Ultsch, A., Thies, M., and Lohken, I. (2006). Modeling timbre distance with temporal statistics from polyphonic music. *Audio, Speech, and Language Processing, IEEE Transactions on*, 14(1):81– 90.

Müller, M., Charypar, D., and Gross, M. (2003). Particle-based fluid simulation for interactive applications. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 154–159, San Diego, California. Eurographics Association.

Müller, M., Solenthaler, B., Keiser, R., and Gross, M. (2005). Particle-based fluid-fluid interaction. In *Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 237–244, Los Angeles, California. ACM.

Murphy, D., Kelloniemi, A., Mullen, J., and Shelley, S. (2007). Acoustic modeling using the digital waveguide mesh. *Signal Processing Magazine, IEEE*, 24(2):55–66.

Murphy, D., Shelley, S., Beeson, M., Moore, A., and Southern, A. (2008). Hybrid room impulse response synthesis in digital waveguide mesh based room acoustics simulations. In *Proc. of the Int. Conference on Digital Audio Effects (DAFx-08)*.

Narain, R., Kwatra, V., Lee, H., Kim, T., Carlson, M., and Lin, M. (2007). Feature-Guided dynamic texture synthesis on continuous flows. In *Feature-Guided Dynamic Texture Synthesis on Continuous Flows*.

O'Brien, J. F., Cook, P. R., and Essl, G. (2001). Synthesizing sounds from physically based motion. In *Proceedings of ACM SIGGRAPH 2001*, pages 529–536. ACM Press.

O'Brien, J. F., Shen, C., and Gatchalian, C. M. (2002). Synthesizing sounds from rigid-body simulations. In *The ACM SIGGRAPH 2002 Symposium on Computer Animation*, pages 175–181. ACM Press.

Ochmann, M. (1995). The source simulation technique for acoustic radiation problems. *Acta Acustica united with Acustica*, 81(6):512–527.

Ochmann, M. (1999). The full-field equations for acoustic radiation and scattering. *The Journal of the Acoustical Society of America*, 105:2574.

Oppenheim, A. V., Schafer, R. W., and Buck, J. R. (1989). *Discrete-time signal processing*, volume 1999. Prentice hall Englewood Cliffs, NJ:.

Pai, D. K., Doel, K. v. d., James, D. L., Lang, J., Lloyd, J. E., Richmond, J. L., and Yau, S. H. (2001). Scanning physical interaction behavior of 3d objects. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '01, pages 87–96, New York, NY, USA. ACM.

Pampalk, E., Rauber, A., and Merkl, D. (2002). Content-based organization and visualization of music archives. In *Proceedings of the tenth ACM international conference on Multimedia*, pages 570–579. ACM.

Picard, C., Tsingos, N., and Faure, F. (2009). Retargetting example sounds to interactive physics-driven animations. In *AES 35th International Conference-Audio for Games, London, UK*.

Pierce, A. D. (1989). *Acoustics: an introduction to its physical principles and applications*. Acoustical Society of America.

Plesset, M. and Prosperetti, A. (1977). Bubble dynamics and cavitation. 9:145–185.

Popov, M. M. (1982). A new method of computation of wave fields using gaussian beams. *Wave motion*, 4(1):85–97.

Porter, M. B. and Bucker, H. P. (1987). Gaussian beam tracing for computing ocean acoustic fields. *The Journal of the Acoustical Society of America*, 82(4):1349–1359.

Pozrikidis, C. (2004). Three-dimensional oscillations of rising bubbles. *Engineering Analysis with Boundary Elements*, 28(4):315–323.

Prosperetti, A. and Oguz, H. (1993). The impact of drops on liquid surfaces and the underwater noise of rain. 25:577–602.

Pumphrey, H. C. and Elmore, P. A. (1990). The entrainment of bubbles by drop impacts. *Journal of Fluid Mechanics*, 220:539–567.

Quatieri, T. and McAulay, R. (1985). Speech transformations based on a sinusoidal representation. In *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP '85.*, volume 10, pages 489 – 492.

Raghuvanshi, N. and Lin, M. (2006). Symphony: Real-time physically-based sound synthesis. In *Proceedings of Symposium on Interactive 3D Graphics and Games*.

Raghuvanshi, N., Narain, R., and Lin, M. C. (2009a). Efficient and accurate sound propagation using adaptive rectangular decomposition. *IEEE Trans. Visualization and Computer Graphics*, 15(5):789–801.

Raghuvanshi, N., Narain, R., and Lin, M. C. (2009b). Efficient and Accurate Sound Propagation Using Adaptive Rectangular Decomposition. *IEEE Transactions on Visualization and Computer Graphics*, 15(5):789–801.

Raghuvanshi, N., Snyder, J., Mehra, R., Lin, M., and Govindaraju, N. (2010). Precomputed wave simualtion for real-time sound propagation of dynamic sources in complex scenes. *ACM Trans. on Graphics (Proc. of ACM SIGGRAPH)*, 29(3).

Rayleigh, L. (1917). On pressure developed in a liquid during the collapse of a spherical cavity. *Philosophical Magazine*.

Ren, Z., Yeh, H., and Lin, M. (2010). Synthesizing contact sounds between textured models. In *Virtual Reality Conference (VR), 2010 IEEE*, pages 139 –146.

Ren, Z., Yeh, H., and Lin, M. C. (2012). Geometry-Invariant Material Perception: Analysis and Evaluation of Rayleigh Damping Model. *UNC Technical Report*.

Roads, C. (2004). *Microsound*. The MIT Press.

Robinson-Mosher, A., Shinar, T., Gretarsson, J., Su, J., and Fedkiw, R. (2008). Two-way coupling of fluids to rigid and deformable solids and shells. *ACM Trans. Graph.*, 27(3):1–9.

Sakamoto, S., Seimiya, T., and Tachibana, H. (2002). Visualization of sound reflection and diffraction using finite difference time domain method. *Acoustical Science and Technology*, 23(1):34–39.

Sakamoto, S., Ushiyama, A., and Nagatomo, H. (2006). Numerical analysis of sound propagation in rooms using the finite difference time domain method. *The Journal of the Acoustical Society of America*, 120(5):3008.

Sakamoto, S., Yokota, T., and Tachibana, H. (2004). Numerical sound field analysis in halls using the finite difference time domain method. In *RADS 2004*, Awaji, Japan.

Salomons, E. M. (2001). *Computational atmospheric acoustics*. Springer.

Savioja, L. (1999). *Modeling Techniques for Virtual Acoustics*. Doctoral thesis, Helsinki University of Technology, Telecommunications Software and Multimedia Laboratory, Report TML-A3.

Scavone, G. P. and Cook, P. R. (1998). Real-time computer modeling of woodwind instruments. Woodbury, NY. Acoustical Society of America.

Serra, X. (1989). *A System for Sound Analysis/Transformation/Synthesis based on a Deterministic plus Stochastic Decomposition*. PhD thesis.

Serra, X. (1997). Musical sound modeling with sinusoids plus noise. *Musical signal processing*, pages 497–510.

Serra, X. and Smith III, J. (1990). Spectral modeling synthesis: A sound analysis/synthesis system based on a deterministic plus stochastic decomposition. *Computer Music Journal*, 14(4):12–24.

Shabana, A. (1997). *Vibration of discrete and continuous systems*. Springer Verlag.

Sirignano, W. A. (2000). Fluid dynamics and transport of droplets and sprays. *Journal of Fluids Engineering*, 122(1):189–190.

Southern, A., Siltanen, S., and Savioja, L. (2011). Spatial room impulse responses with a hybrid modeling method. In *Audio Engineering Society Convention 130*.

Stam, J. (1999). Stable fluids. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pages 121–128. ACM Press/Addison-Wesley Publishing Co.

Steiglitz, K. and McBride, L. (1965). A technique for the identification of linear systems. *Automatic Control, IEEE Transactions on*, 10(4):461–464.

Svensson, U. P., Fred, R. I., and Vanderkooy, J. (1999). An analytic secondary source model of edge diffraction impulse responses. *J. Acoustical Society of America*, 106(5):2331–2344.

Taflove, A. and Hagness, S. C. (2005). *Computational Electrodynamics: The Finite-Difference Time-Domain Method*. Artech House.

Taylor, M., Chandak, A., Antani, L., and Manocha, D. (2009). Resound: Interactive sound rendering for dynamic virtual environments. In *Proc. ACM Multimedia*.

Taylor, M., Chandak, A., Qi Mo, Lauterbach, C., Schissler, C., and Manocha, D. (2012). Guided multiview ray tracing for fast auralization. *Visualization and Computer Graphics, IEEE Transactions on*, 18(11):1797–1810.

Thompson, L. L. (2006). A review of finite-element methods for time-harmonic acoustics. *J. Acoustical Society of America*, 119(3):1315–1330.

Thürey, N., Müller-Fischer, M., Schirm, S., and Gross, M. (2007a). Real-time breaking waves for shallow water simulations. In *Proceedings of the 15th Pacific Conference on Computer Graphics and Applications*, pages 39–46. IEEE Computer Society.

Thürey, N., Sadlo, F., Schirm, S., Müller-Fischer, M., and Gross, M. (2007b). Real-time simulations of bubbles and foam within a shallow water framework. In *Proceedings of the 2007 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 191–198, San Diego, California. Eurographics Association.

Tolstoy, A. (1996). 3-d propagation issues and models. *Journal of Computational Acoustics*, 4(03):243–271.

Trebien, F. and Oliveira, M. (2009). Realistic real-time sound re-synthesis and processing forinteractive virtual worlds. *The Visual Computer*, 25:469–477.

Tsingos, N. (2009). Pre-computing geometry-based reverberation effects for games. *35th AES Conference on Audio for Games*.

Tsingos, N., Funkhouser, T., Ngan, A., and Carlbom, I. (2001). Modeling acoustics in virtual environments using the uniform theory of diffraction. In *Proc. SIGGRAPH 2001*, pages 545–552.

Välimäki, V., Huopaniemi, J., Karjalainen, M., and Jánosy, Z. (1996). Physical modeling of plucked string instruments with application to real-time sound synthesis. *Journal of the Audio Engineering Society*, 44(5):331–353.

Välimäki, V. and Tolonen, T. (1997). Development and calibration of a guitar synthesizer. *PREPRINTS-AUDIO ENGINEERING SOCIETY*.

van den Doel, K. (2005). Physically based models for liquid sounds. *ACM Trans. Appl. Percept.*, 2(4):534–546.

van den Doel, K., Knott, D., and Pai, D. K. (2004). Interactive simulation of complex audiovisual scenes. *Presence: Teleoper. Virtual Environ.*, 13:99–111.

van den Doel, K., Kry, P., and Pai, D. (2001). FoleyAutomatic: physically-based sound effects for interactive simulation and animation. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 537–544. ACM New York, NY, USA.

van den Doel, K. and Pai, D. K. (1998). The sounds of physical shapes. *Presence: Teleoper. Virtual Environ.*, 7:382–395.

van den Doel, K. and Pai, D. K. (2002a). Measurements of perceptual quality of contact sound models. In *Proc. of the International Conference on Auditory Display (ICAD 2002)*, pages 345—349, Kyoto, Japan.

van den Doel, K. and Pai, D. K. (2002b). Measurements of perceptual quality of contact sound models. In *In Proceedings of the International Conference on Auditory Display (ICAD 2002*, pages 345–349.

Van Duyne, S. and Smith, J. O. (1993). The 2-d digital waveguide mesh. In *Applications of Signal Processing to Audio and Acoustics, 1993. Final Program and Paper Summaries., 1993 IEEE Workshop on*, pages 177–180.

Červený, V. (1985). Ray synthetic seismograms for complex two-and three-dimensional structures. *J. Geophys*, 58:2–26.

Červený, V. (2000). Summation of paraxial gaussian beams and of paraxial ray approximations in inhomogeneous anisotropic layered structures. *Seismic waves in complex 3-D structures, Report*, 10:121–59.

Červený, V. (2005). *Seismic ray theory*. Cambridge University Press.

Červený, V., Klimeš, L., and Pšenčík, I. (1988). Complete seismic-ray tracing in three-dimensional structures. *Seismological Algorithms, Academic Press, New York*, pages 89–168.

Vladimirov, V. S. (1976). Generalized functions in mathematical physics. *Moscow Izdatel Nauka*, 1.

Vorlander, M. (1989). Simulation of the transient and steady-state sound propagation in rooms using a new combined ray-tracing/image-source algorithm. *J. Acoustical Society of America*, 86(1):172–178.

Wang, S., Sekey, A., and Gersho, A. (1992). An objective measure for predicting subjective quality of speech coders. *IEEE Journal on Selected Areas in Communications*, 10(5):819–829.

Wang, Y., Safavi-Naeini, S., and Chaudhuri, S. (2000). A hybrid technique based on combining ray tracing and FDTD methods for site-specific modeling of indoor radio wave propagation. *Antennas and Propagation, IEEE Transactions on*, 48(5):743 –754.

Waterman, P. C. (2009). T-matrix methods in acoustic scattering. *The Journal of the Acoustical Society of America*, 125(1):42–51.

Weyl, H. (1919). Ausbreitung elektromagnetischer wellen ber einem ebenen leiter. *Annalen der Physik*, 365(21):481–500.

Yee, K. (1966). Numerical solution of inital boundary value problems involving maxwell's equations in isotropic media. *Antennas and Propagation, IEEE Transactions on [legacy, pre - 1988]*, 14(3):302–307.

Zheng, C. and James, D. L. (2009). Harmonic fluids. In *ACM SIGGRAPH 2009 papers*, pages 1–12, New Orleans, Louisiana. ACM.

Zheng, C. and James, D. L. (2010). Rigid-body fracture sound with precomputed soundbanks. *ACM Trans. Graph.*, 29:69:1–69:13.

Zheng, C. and James, D. L. (2011). Toward high-quality modal contact sound. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2011)*, 30(4).

Zienkiewicz, O. C., Taylor, R. L., and Nithiarasu, P. (2006). *The finite element method for fluid dynamics*. Butterworth-Heinemann, 6 edition.

Ziolkowski, R. W. and Deschamps, G. A. (1980). *The Maslov method and the asymptotic Fourier transform: Caustic analysis*.

Zwicker, E. and Fastl, H. (1999). *Psychoacoustics: Facts and models*, volume 254. Springer New York, 2nd updated edition edition.